21世纪高等学校规划教材 | 软件工程





高级软件测试管理

郑文强　周震漪　马均飞　编著

清华大学出版社

21 世纪高等学校规划教材 • 软件工程

高级软件测试管理

郑文强　周震漪　马均飞　编著

清 华 大 学 出 版 社
北　京

内 容 简 介

本书是国际软件测试认证委员会（ISTQB）大中华区分会 CSTQB 指定的高级测试经理（即 ISTQB –TM 模块）认证考试官方培训推荐教材。同时，本书也可以作为高校软件工程及相关专业软件测试课程的理想教材，以及作为测试工程师、测试经理和测试过程改进人员等的重要学习参考资料。

本书从测试过程、测试管理、评审、缺陷管理、测试过程改进、测试工具及自动化，以及人员管理等几个方面，全面介绍了测试经理在测试过程中需要关注的流程、测试文档、管理技术与方法、测试管理工具与自动化等内容。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。侵权举报电话：010-62782989　13701121933

图书在版编目（CIP）数据

高级软件测试管理 / 郑文强，周震漪，马均飞编著. —北京：清华大学出版社，2017

（21 世纪高等学校规划教材・软件工程）

ISBN 978-7-302-46710-6

Ⅰ. ①高…　Ⅱ. ①郑…　②周…　③马…　Ⅲ. ①软件–测试　Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字（2017）第 038672 号

责任编辑： 魏江江　薛　阳
封面设计：
责任校对： 焦丽丽
责任印制：

出版发行： 清华大学出版社
网　　址： http://www.tup.com.cn, http://www.wqbook.com
地　　址： 北京清华大学学研大厦 A 座　　**邮　　编：** 100084
社 总 机： 010-62770175　　**邮　　购：** 010-62786544
投稿与读者服务： 010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈： 010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：
经　　销： 全国新华书店
开　　本： 185mm×260mm　　**印　　张：** 18　　**字　　数：** 437 千字
版　　次： 2017 年 6 月第 1 版　　**印　　次：** 2017 年 6 月第 1 次印刷
印　　数：
定　　价： 　　元

产品编号：066952-01

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施"高等学校本科教学质量与教学改革工程（简称'质量工程'）"，通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部"质量工程"的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合"质量工程"的实施，满足各高校教学质量和教学改革的需要。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作，提高教学质量的若干意见》精神，紧密配合教育部已经启动的"高等学校教学质量与教学改革工程精品课程建设工作"，在有关专家、教授的倡议和有关部门的大力支持下，我们组织并成立了"清华大学出版社教材编审委员会"（以下简称"编委会"），旨在配合教育部制定精品课程教材的出版规划，讨论并实施精品课程教材的编写与出版工作。"编委会"成员皆来自全国各类高等学校教学与科研第一线的骨干教师，其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求，"编委会"一致认为，精品课程的建设工作从开始就要坚持高标准、严要求，处于一个比较高的起点上；精品课程教材应该能够反映各高校教学改革与课程建设的需要，要有特色风格、有创新性（新体系、新内容、新手段、新思路，教材的内容体系有较高的科学创新、技术创新和理念创新的含量）、先进性（对原有的学科体系有实质性的改革和发展，顺应并符合 21 世纪教学发展的规律，代表并引领课程发展的趋势和方向）、示范性（教材所体现的课程体系具有较广泛的辐射性和示范性）和一定的前瞻性。教材由个人申报或各校推荐（通过所在高校的"编委会"成员推荐），经"编委会"认真评审，最后由清华大学出版社审定出版。

目前，针对计算机类和电子信息类相关专业成立了两个“编委会”，即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。推出的特色精品教材包括：

（1）21 世纪高等学校规划教材 • 计算机应用——高等学校各类专业，特别是非计算机专业的计算机应用类教材。

（2）21 世纪高等学校规划教材 • 计算机科学与技术——高等学校计算机相关专业的教材。

（3）21 世纪高等学校规划教材 • 电子信息——高等学校电子信息相关专业的教材。

（4）21 世纪高等学校规划教材 • 软件工程——高等学校软件工程相关专业的教材。

（5）21 世纪高等学校规划教材 • 信息管理与信息系统。

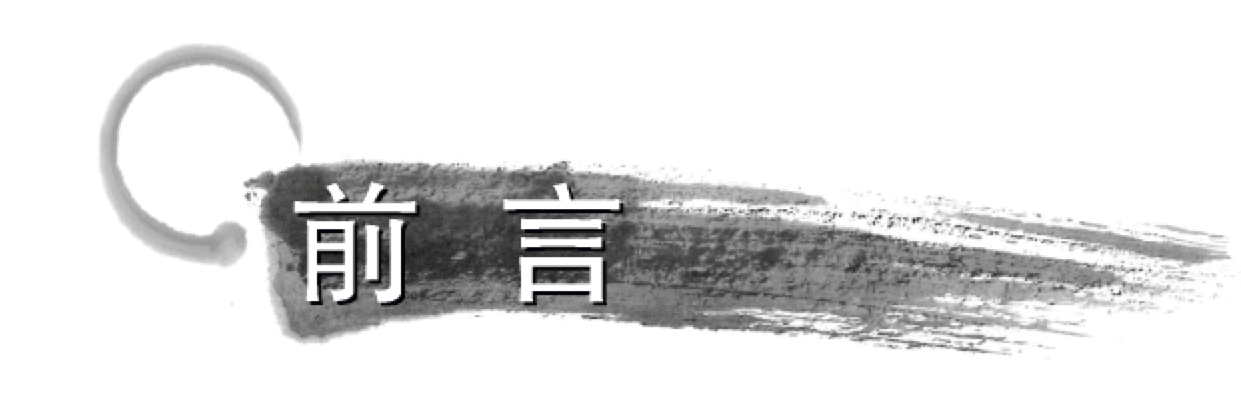
（6）21 世纪高等学校规划教材 • 财经管理与应用。

（7）21 世纪高等学校规划教材 • 电子商务。

（8）21 世纪高等学校规划教材 • 物联网。

清华大学出版社经过三十年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会
联系人：魏江江
E-mail:weijj@tup.tsinghua.edu.cn



前言

1. ISTQB 简介

ISTQB 目前拥有 54 个分会，覆盖包括美国、德国、英国、法国、印度等在内的 110 多个国家和地区。来自于这些国家和地区的数百位测试领域专家作为志愿者服务于 ISTQB 及其倡导的软件测试工程师认证体系。截至目前在全球范围内参加过 ISTQB 认证的软件测试工程师已达到 650 000 人，并每季度以超过 20 000 人的速度递增，使得 ISTQB 为测试行业的第一大认证机构，在整个 IT 行业居第三位（仅次于 PMI 和 ITIL）。

CSTQB（Chinese Software Testing Qualifications Board）全权代表 ISTQB 授权在大中华区域内推广 ISTQB 软件测试工程师认证体系，认证、管理培训机构和考试机构，接受 ISTQB 全面的业务指导和授权。

2. 编写目的

目前国内对软件测试的重视程度在不断提高，软件企业对测试工程师的测试技能要求也在不断提高。随着通过 ISTQB 基础级认证的测试工程师越来越多，他们将会渴望获得更高层次的测试知识和技能，因此 ISTQB 高级测试经理认证就是他们测试职业规划中的一个重要发展方向。

目前，国内针对 ISTQB 高级测试经理认证的主要参考资料是 ISTQB 高级测试经理认证大纲，包括英文版本和中文版本。由于大纲提供的是概要性的测试知识描述，无法有效地帮助学员进行 ISTQB 高级测试经理认证的考前学习和复习。为了帮助参加 ISTQB 高级测试经理认证考试的学员系统学习测试管理知识，以帮助测试人员尽快掌握国际通用的软件测试管理知识和技能，同时推动国内软件测试行业的国际化和标准化，本书作者合作一起编写了这本 ISTQB 高级认证的参考书。本书完全覆盖了 ISTQB 高级测试经理认证大纲的内容，同时在每个章节中罗列了相关的学习目标和测试术语，而且每个章节后面提供了针对学习目标的模拟题和参考答案，以方便测试人员更好地进行复习和学习。

3. 本书结构

本书共 7 个章节，以软件测试过程为基础，描述了软件测试过程中每个测试阶段涉及的主要测试管理活动、管理技术与方法、测试管理工具、测试过程改进和角色与职责等内容。每个章节的主要内容如下。

第 1 章 测试过程：介绍了测试过程的几大阶段，如测试计划与监控、测试分析、测试设计、测试实施、测试执行、测试出口准则评估与测试总结报告，以及测试结束活动。测试经理主要关注在测试计划、测试监视和控制、评估和报告，以及测试回顾总结和改进等方面。

第 2 章 测试管理：主要介绍特定背景下如何开展测试管理活动（例如测试干系人、

其他开发生命周期活动及工作产品、测试与开发的集成等)、基于风险测试过程中进行测试优先级设定和工作量分配、管理测试过程中的测试文档、有效开展测试估算、定性与定量分析测试的商业价值，以及阐述分布式测试、外包测试和内包测试的特点和优缺点，并介绍了测试过程中可能涉及的各种国际、国内和行业的测试相关标准。

第 3 章 评审：主要介绍了管理评审和审计的特点与区别、评审过程每个阶段测试经理需要关注的检查点、需要收集针对评审产品和评审过程相关的度量，以及如何对评审过程进行管理。

第 4 章 缺陷管理：主要介绍了缺陷的管理生命周期，包括缺陷管理流程和状态、管理无效和重复缺陷的策略、跨职能缺陷管理以改善缺陷管理过程的效率和有效性；同时也介绍了缺陷报告的主要内容和元素，以及如何根据缺陷报告提供的信息对开发过程和测试过程进行评估。

第 5 章 改进测试过程：主要阐述了戴明改进循环 PDCA、改进测试过程 IDEAL 的主要阶段和活动、角色和职责等。同时，简单介绍了不同的改进模型：TMMi、TPI-Next、CTP 和 STEP，以及它们各自的特点、成熟度级别、关键域等。

第 6 章 测试工具及自动化：主要介绍了开源工具和定制工具的特点和优缺点、投资回报分析中的一次性成本和经常性成本组成，以及选择测试工具的流程。另外，也描述了工具的生命周期和评估工具的适合度量。

第 7 章 人员管理：主要介绍了测试人员的测试技能要求，并根据测试过程中涉及的技能要求，对测试工程师进行全面评估，并制定相应得到培训计划。同时描述了测试团队需要具备的硬技能和软技能，选择合适的测试独立性以满足测试组织要求。最后描述了如何高效开展团队内的激励和沟通。

作者分工

本书作者郑文强、周震漪和马均飞共同承担了本书的编写和评审工作，他们是国内最早参与 ISTQB 活动的 CSTQB 专家组成员，也是国内最早获得 ISTQB 基础级认证证书和高级认证证书的成员之一。本书作者有总共超过 50 年的测试工作相关经验，他们对 ISTQB 软件测试管理知识体系的深入理解和学习目标的诠释，确信可以为读者带来不一样的感受。

致谢

本书的出版离不开在我们成长过程中给予我们帮助的同学、同事和朋友，他们为此书的出版提供了诚恳的指导和宝贵的意见。同时，特别感谢 CSTQB 办公室对本书编写的大力支持。

感谢清华大学出版社魏江江主任为本书出版提供大力支持，本书才得以在这么短的时间内与大家见面；同时感谢出版社其他人员，他们的专业素质和敬业精神令我们感动。

最后要感谢我们的家人，这本书的写作占用了大量本该陪伴家人的晚上和周末时间，没有她们的支持和鼓励，这本书也很难和大家见面。

郑文强

2017 年 1 月


目 录

第1章 测试过程

本章学习目标如表 1-1 所示。

表 1-1 学习目标

编　号	学习目标描述	级别
TM-1.2.1	为了计划测试活动和工作产品来实现测试目标，必须对一个系统的测试需求进行分析	K4
TM-1.3.1	使用可追溯性来检查与测试目标、测试策略和测试计划相关的已定义测试条件的完整性和一致性	K3
TM-1.3.2	解释可能影响所规定测试条件详细程度的因素，以及详细规定测试条件的优点和缺点	K2
TM-1.4.1	使用可追溯性来检查与已定义测试条件相关的所设计测试用例的完整性和一致性	K3
TM-1.5.1	使用风险、优先级、测试环境和数据依赖以及限制条件来制定测试执行的进度，该进度针对测试目标、测试策略和测试计划是完整和一致的	K3
TM-1.6.1	使用可追溯性来监督测试进展与测试目标、测试策略和测试计划的一致性和完整性	K3
TM-1.7.1	解释在测试过程中准确和及时信息收集的重要性，以便支持准确的报告和对照出口准则进行评价	K2
TM-1.8.1	总结 4 组测试结束活动	K2
TM-1.8.2	实现项目回顾以评价过程和发现改进领域	K3

本章相关术语如表 1-2 所示。

表 1-2 术语

英　文	中　文	说　明
exit criteria	出口准则	通过与利益干系人达成一致的一组通用和特定的条件，正式允许一个过程结束。设置出口准则的目的在于防止将没有完成的任务错误地看成已经完成。测试中使用出口准则来报告和计划什么时候可以停止测试（与 Gilb and Graham 一致）
test case	测试用例	为特定目标或测试条件（例如，执行特定的程序路径，或是验证与特定需求的一致性）而制定的一组输入值、执行入口条件、预期结果和执行出口条件（与 IEEE 610 一致）[GBT 11457]
test closure	测试结束	从已完成的测试活动中收集数据，总结基于测试件及相关事实和数据的测试结束阶段，包括对测试件的最终处理和归档，以及测试过程评估（包含测试评估报告的准备）。参见 test process
test condition	测试条件	组件/系统中能被一个或多个测试用例验证的条目或事件。例如，功能、事务、特性、质量特性或者结构化元素
test control	测试控制	当监测到与预期情况背离时,制定和应用一组修正动作以使测试项目保持正常进行的测试管理工作。参见 test management

续表

英　　文	中　　文	说　　明
test design	测试设计	（1）参见 test design specification。 （2）将测试目标转换成具体的测试条件和测试用例的过程[GBT 11457]
test execution	测试执行	对被测组件/系统执行测试，产生实际结果的过程
test implementation	测试实施	开发、排序测试规程，创建测试数据，必要时还包括准备测试用具和编写自动化测试脚本的过程
test log	测试日志	按时间顺序排列的有关测试执行所有相关细节的记录[GBT 9386]
test planning	测试计划	描述预期测试活动的范围、方法、资源和进度的文档。它标识了测试项、需测试的特性、测试任务、任务负责人、测试人员的独立程度、测试环境、测试设计技术、测试的进入和退出准则和选择的合理性、需要紧急预案的风险，是测试策划过程的一份记录（与 IEEE Std 829 一致）[GBT 9386]
test procedure	测试规程	参见 test procedure specification[GBT 11457]
test script	测试脚本	通常指测试规程说明，尤其对自动化测试[GBT11457]
test summary report	测试总结报告	总结测试活动和结果的文档。也包括对测试项是否符合退出准则进行的评估[GBT 9386]

1.1 简介

软件测试贯穿整个软件开发生命周期，是与软件开发并行的一个完整的过程，测试的尽早介入是软件测试的基本原则。将软件测试看作是软件开发的一个阶段，或仅仅是通过运行软件进行的检查活动，这不是系统化测试的理念。为了有效实现软件测试各个维度的测试目标，需要和开发过程一样定义一个正式而完整的软件测试过程，即涉及各个测试阶段、活动、技术、文档等内容的过程，来指导和管理软件测试活动，以提高测试效率、测试有效性和测试质量，同时不断改进开发过程和测试过程。

作为广义的软件测试过程，ISTQB 基础级认证大纲中定义了完整的软件测试过程，覆盖了测试过程中所有的测试阶段和活动，例如，什么时候应该做什么、应该检查什么、输出什么等，即该过程可以帮助测试人员更好地了解测试活动和任务。其定义的测试过程包含下面 5 个阶段。

（1）测试计划和控制；

（2）测试分析和设计；

（3）测试实施和执行；

（4）评估出口准则和报告；

（5）测试结束活动。

为了更好地适应软件开发生命周期的要求和有效地进行测试监控，ISTQB 高级测试经理大纲将测试过程中的一些活动独立出来进行讲解。分解之后的测试过程如下。

（1）计划、监督和控制；

（2）分析；

（3）设计；
（4）实施；
（5）执行；
（6）评估出口准则和报告；
（7）测试结束活动。

上述的测试过程给人的感觉好像各个阶段是顺序进行的，但是实际上有些测试阶段在时间上是可以有重叠的，甚至是可以并行进行的，例如，测试分析和设计、测试实施和执行阶段在时间上是可以有部分重叠的，而测试控制活动会贯穿于整个测试过程。

根据软件产品的特点、开发模型、团队能力等的不同，测试团队需要对上述的测试过程进行合理的裁剪。ISTQB 基础级认证大纲定义的测试过程可能和每个实际软件项目所定义的测试过程都存在一些不同，但其中的测试活动和测试任务都会在测试中发挥重要的作用。

另外，测试经理和测试人员在测试过程中的职责会有所不同，本章节将主要关注点放在与测试经理相关的测试活动上。

1.2 测试计划、监督与控制

测试计划阶段的主要活动包括：识别测试目标（例如，发现缺陷、增加信心、为决策提供信息，以及缺陷预防等）、测试任务，以及为实现测试目标和任务而确定的必要的测试活动和资源。

测试监督阶段的主要活动包括：对测试过程中的各个测试活动进行监视，通过收集和分析测试活动的各种度量数据，以评估当前项目和测试的状态，例如，测试进度状况、测试质量状况等。

测试控制阶段的主要活动包括：通过将测试监督获取的测试数据和状态，与测试计划中的目标和要求进行比较，假如两者之间出现偏差（大于测试计划中定义的偏差范围或阈值），测试经理需要采取必要的措施纠正偏差，以满足测试任务和目标要求。

不管是测试计划、监督还是控制活动都应该是持续进行的活动，即需要贯穿于整个测试生命周期。测试计划、监督与控制活动是测试经理的主要职责。

1.2.1 测试计划

软件项目启动时就需要进行测试计划相关的活动，并贯穿整个软件测试生命周期，直到完成测试结束活动。制定测试计划的目的是通过确定测试任务、定义测试对象和详细的测试活动来达到组织的目标和使命。针对不同的测试级别，例如，集成测试或系统测试，需要制定不同的测试计划，并且文档化。

为了满足测试策略（详细内容请参考 2.4.2 节）中定义的测试目标和测试任务，测试计划制定过程中需要识别所需的测试活动和测试资源。测试计划也包括识别、收集和跟踪测试度量项的方法，这些度量项将用于指导软件项目、确定与测试计划之间的符合程度，以及评估是否达到定义的测试目标等。测试计划阶段确定合理有用的度量项，可以有效地

帮助测试团队选择测试工具、安排培训和构建测试文档指南。

测试计划的制定需要考虑各种因素的影响，例如，组织的质量方针和测试方针、测试范围、测试目标、测试相关的风险、测试约束条件和资源的可用性等。随着软件项目的不断推进和深入，测试经理将获得更多的项目信息和具体细节，并基于这些信息对测试计划的内容进行更新，即测试计划的制定是个持续的过程，需要在整个测试过程中进行不断的调整和更新，例如，根据从测试活动中得到的反馈信息，可以识别测试过程中存在的新的风险，从而对测试计划中的风险部分内容做相应的修改。

下面对制定测试计划相关的测试活动做个简单的解释。

1．选择测试策略

测试策略描述了组织或软件项目采用的测试方法，包括产品风险和项目风险管理、测试级别和阶段定义，以及确定测试的概要活动等。根据测试活动开始时间的不同，测试策略可以分为预防型策略（尽早设计测试用例以预防缺陷）和应对型策略（测试活动在软件产品实现后开始）。详细内容请参考 2.4.2 节。

测试计划中选择的测试策略，有助于测试经理确定在测试过程中应进行的测试任务。例如，假如采用基于风险的测试策略（详细内容参考 2.3 节），测试经理和测试人员可以通过风险分析指导测试过程中的风险识别、风险缓解、风险评估等活动，以降低产品风险，并帮助制定风险的应急措施。假如在测试过程中发现多个严重程度高的安全性相关的问题，则测试团队应花费更多的工作量设计和执行安全性测试。同样，假如发现软件产品的设计规格说明中存在严重缺陷，则在测试过程中可以安排合适的人员参与该文档的评审。

2．确定测试优先级

软件测试的一条基本原则是：穷尽测试是不可能的。因此，测试经理必定面临如何在有限时间和资源的情况下开展测试活动的问题。另外，测试团队经常面临的一个问题是：开发团队经常会延迟交付软件版本给测试团队，从而导致测试团队只能压缩测试时间以满足软件产品版本及时发布的困境。此时，将测试任务和测试活动进行优先级的划分，并将测试资源分配到最重要的地方是测试经理可以采取的一种有效方式。

测试过程中可以采用多种不同的测试技术，帮助测试经理确定测试活动和任务的优先级，例如，基于风险的测试策略（详细内容请参考 2.3 节）、基于需求的测试策略等。假如采用基于风险的测试策略，可以尽早识别软件产品的产品风险，并基于该风险的可能性和严重程度得到的风险级别确定不同测试活动和任务的优先级。例如，当软件产品的性能的风险级别较高，测试团队在获得软件版本之后，尽快进行性能测试。而基于需求的测试策略，被测对象的测试依据中定义的需求优先级，可以作为测试活动和优先级安排的重要参考。类似地，如果测试过程中采用了应对型的测试策略（例如：探索性测试），测试团队必须尽早创建测试章程以确定测试范围和内容（详细内容请参考 2.2.5 节），以及准备动态测试所需的测试工具。

3．定义测试方法

测试经理在测试计划阶段应清楚地定义测试方法，包括采用的测试级别、每个测试级别的测试目标以及每个测试级别使用的测试技术。例如，静态测试（评审和静态分析）在软件开发生命周期的早期使用得比较多；针对安全关键的软件产品（例如，航空航天相关的软件产品），假如其采用的是基于风险的测试策略，需要根据风险评估结果确定不同测试

级别需要达到的代码覆盖率要求，以及针对不同测试级别可以采用的测试技术。

4．确定测试文档关系

测试活动并不是孤立存在的，在测试依据（即能够从中推断出被测软件产品/组件需求的所有文档，而测试用例正是基于测试依据开展分析、设计和验证的）、测试条件（能够通过一个或多个测试用例进行验证的条目或事件，例如，业务功能、质量特性或被测组件结构元素等，即测试条件决定测试什么）和覆盖测试条件的测试用例之间可能存在复杂的关系，并且这些工作产品之间常常存在一对多或多对多的关联。

测试计划阶段必须明确测试依据、测试条件和测试用例之间的对应关系，测试经理才能基于上述关系选择合理的测试工具，以及基于上述对应关系进行后续的测试监督和控制。例如，建立从测试条件到软件产品需求之间的可追溯性，不仅有助于在需求发生变更时的影响分析，也可以帮助测试经理在测试执行过程中进行合理的需求覆盖率分析。

另外，测试计划需要确定测试过程需要输出的测试文档类型，例如，测试计划文档、测试设计规格说明等。测试过程中建立统一的测试文档格式，有助于测试经理或者测试人员编写相关文档的效率，同时统一测试文档格式，也可以提高文档编写的质量，避免文档内容的缺失。详细的测试文档模板可以参考 IEEE Std 829—2008 标准，关于测试管理文档部分，可以参考 2.4 节的内容。

5．确定开发工作产品和测试工作产品之间的关系

开发团队与测试团队在软件开发生命周期中输出的工作产品之间也存在关联，例如，可追溯性矩阵需要跟踪设计人员的详细规格说明、业务分析人员的业务需求与测试团队定义的测试工作产品之间的关系。例如，假如开发过程采用典型 V 模型，测试经理可以在测试计划阶段明确：在正式设计和实施详细测试用例之前，开发团队的详细设计规格说明需要经过评审并得到批准。而遵循敏捷开发生命周期时，开发团队与测试团队之间需要更多地采用更紧密的信息交流和沟通方式。

测试经理制定测试计划时，必须以软件工作产品作为重要输入和基础。软件开发生命周期中主要的软件工作产品包括：项目合同、产品范围说明、项目计划文档、用户需求规格说明、系统需求规格说明、系统设计规格说明、模块设计规格说明等。上述软件工作产品在下面的测试活动中发挥着重要的作用。

（1）开展测试估算。测试工作量的估算依赖于开发过程中提供的工作产品，例如，软件产品范围说明和系统需求规格说明等文档。同时，软件产品特点、产品类型、质量要求和团队能力等都会影响测试工作量的估算。

（2）选择测试方法。根据软件产品的内容和说明，选择合适的测试方法验证系统需求和产品功能。因此，软件产品的特征和内容是选择合适的测试方法和技术的基础。

（3）计划测试资源。根据软件产品特点和测试范围，合理选择和安排测试资源，例如，合适的测试人员、测试环境和设备等。

（4）制定测试执行进度计划。根据上面的这些内容制定详细的测试执行进度计划。

6．确定测试范围

针对不同的测试级别，其测试关注的范围和重点是不一样的。因此不同测试级别应该列出本次计划需要覆盖的测试内容，例如，重点覆盖性能测试、关注软件产品不同组件之间的接口和交互功能等。同时，测试计划中也需要明确说明不在本次测试范围的产品特性。

根据测试工作产品与软件工作产品之间的关联度，以及测试系统化的要求，将测试范围的每个产品特性与相应的测试规格说明进行对应。确定测试范围时，根据风险分析的结果确定测试重点和优先级，是测试经理不错的选项。

7．确定测试环境

测试环境（例如，测试平台和测试工具）的规划，应该在测试计划阶段考虑，而不是在测试执行阶段才进行规划和实施。测试经理应该在测试计划阶段，与软件产品架构师共同确定初始的测试环境规格说明，验证所需测试资源的可获取性，确保测试环境配置管理人员及时到位负责这些工作，以及了解完成和交付测试环境所需的工作量和时间进度表。

测试环境的范围非常广泛，包括软件产品运行的硬件配置（例如，对于通信系统，可能会使用不同的硬件机架、不同的控制模块、不同的用户模块等）、外部设备的型号规格、网络配置等。这些测试平台相关的硬件应尽可能与用户实际使用中的硬件一致。同时，使用的软件产品的配置、版本、性能应尽可能与用户使用的一致。表 1-3 是通信产品系统测试中环境配置清单的一个模板。

表 1-3　环境配置需求清单

类　别	序　号	数　量	设备名称	配置要求	使用时间
服务器					
PC 终端					
软件					
网络要求					
数据线					
网络线					

测试工具主要是指测试执行过程中需要运用的各种工具，例如，对于复杂的通信产品系统测试，主要包括用户数据发生器和模拟器、协议分析仪、自动化测试的工具、测试管理工具（例如，缺陷管理工具、测试进度管理工具等）等。

对于所有上面提及的测试环境方面的要求，都需要在测试计划阶段进行初步规划。而在实际测试过程中，测试计划阶段常常会忽视测试环境的规划，甚至直到测试执行阶段才被考虑。测试执行阶段才考虑测试环境的规划，可能会给测试执行带来很大的风险，例如，测试执行过程中用到的硬件设备和测试工具，需要购买或者租用时，此时必须尽早对测试环境进行计划，避免影响测试执行的进度和质量。

8．了解外部依赖关系

测试经理应在测试计划阶段识别所有的外部依赖关系和相关的服务级别协议（Service Level Agreements，SLA）。依赖关系包括对外部组织的资源请求、与其他软件项目的依赖关系（例如，开发团队发布软件版本的时间节点与测试团队测试任务安排时间节点的依赖关系）、与外部测试设备供应商或者合作伙伴的依赖关系、与其他软件部署团队以及数据库管理人员之间的依赖关系等。

9．定义测试出口准则

定义测试出口准则也是测试计划中的一个重要内容，其目的是确定什么时候可以结束测试。例如，判断某个测试级别是否可以结束，需要判断当前的测试是否达到了规定的要求。测试出口准则可以基于下面的一些度量进行判断。

（1）测试完整性度量，例如，代码、功能或风险的测试覆盖率；

（2）缺陷密度、缺陷趋势或系统可靠性的度量；

（3）继续测试的成本度量，评估继续测试的收益和结束测试的风险之间的平衡关系；

（4）基于剩余风险的度量，例如，评估没有被修改的缺陷、在某些区域测试覆盖率较低等可能导致的项目风险或产品风险；

（5）软件产品交付时间的度量，例如，客户要求的软件产品交付时间或者测试计划规定的截止时间等。

上面简单地介绍了测试规划过程中测试经理主要关注的测试活动，下面以制定测试执行进度计划为例，说明制定过程中需要考虑的一些因素。

☆示例：制定测试执行进度计划

测试执行进度计划的制定需要兼顾测试范围、测试资源、产品质量、测试时间之间的相互制约关系，并有效达到它们之间的平衡。例如，软件产品发布的时间是确定的，或者根据客户的要求是受到限制的，那么在有限的时间内，需要平衡测试范围、测试资源和产品质量来进行测试执行进度计划的制定。下面阐述在项目实践过程中影响测试执行进度计划制定的主要因素。

1. 时间因素

制定测试执行进度计划首先需要考虑时间因素。对于一些安全关键系统，它们的交付时间受客户的影响比较小，例如，航空航天、医疗软件等，在进度和质量发生冲突的时候，它们更侧重于质量。而对于非安全关键系统，例如，商业软件，由于激烈的市场竞争，组织希望能尽快发布软件产品，或者客户要求的软件产品发布时间已经确定。例如，某软件产品必须在 2016 年 12 月 31 日之前交付给客户，因此，所有的测试活动和开发活动的进度安排都必须围绕这个时间点来进行。

2. 人力因素

测试经理根据组织历史数据和经验，可以估算软件产品的规模和工作量，并基于估算的结果确定测试人员数目和其他的测试资源，例如，在测试执行过程中，该组织测试执行的经验数据是每人每天平均执行 4 个测试用例，如果指定的测试执行时间是 20 个工作日，需要执行的测试用例数目是 400 个，那么从这些数据中可以得到需要的测试人员至少是 5 个（400/（4×20））。

除了测试人员数目的要求，还对测试人员具备的技能水平有所要求。不同的软件产品需要不同技能的测试人员。常见的测试技能要求包括：软件产品使用经验、领域背景知识、测试工具和自动化、测试环境搭建、测试基本理论和技能知识等。测试执行进度计划制定时需要明确测试人员应该具备的技能。如果发现测试团队缺乏相应的技能，需要及时制定招募或培训计划，提高团队成员的相关技能水平。

3. 软件质量

在制定测试执行进度计划的时候，假定 5 个测试人员在 20 天时间内能够完成 400 个测试用例。在实际测试过程中，可能会发现无法完成这个工作，或者至少说非常困难，因为测试计划常常会低估发现缺陷时与开发人员沟通重现缺陷的时间、后续的确认测试和回归测试时间、参加项目例行会议的时间等。因此，在做测试执行进度计划

时，还需要考虑在测试过程中可能发现的缺陷数目，然后根据组织的缺陷方面的验证数据来估算需要花费的时间。软件缺陷的再测试以及回归测试的工作量，在制定测试执行进度计划的时候常常会被遗漏，导致测试后期的测试任务非常繁重。产品的质量、文档的质量、开发过程的成熟度、个人的能力等都会极大地影响测试的进度。

4. 测试管理活动

整个测试过程还包括很多测试管理活动，例如，完成测试执行以后，需要提交测试总结报告，包含测试的内容和范围、测试存在的风险、产品遗留的缺陷以及相应的解决方案和软件质量信息等；需要定期召开测试团队内部会议或参加项目会议等。所以，在测试执行进度计划中也需要包括测试管理相关的工作量。

1.2.2 测试监督和控制

测试经理为了进行高效的测试控制，需要在组织或者项目层面建立监督框架，以便按照测试计划要求跟踪测试工作产品和测试进度的状态，测试监控的框架必须和测试策略中定义的目标一致。测试计划阶段需要定义测试进度与各种监督活动和度量。随着测试活动的不断开展，收集的度量信息可以帮助测试经理判断测试是否满足测试计划的要求。假如测试状态和其他测试进度信息，与制定的测试计划之间出现偏差和问题，测试经理需要采取相应的手段对测试活动进行控制，从而使测试活动能够按照测试计划的要求进行。另外，测试监控得到的反馈数据和信息也有助于更新原先制定的测试计划。

测试控制必须对测试过程中提供的一些信息做出回应，同时做出相应的测试计划的变更。例如，动态测试过程中发现原来认为质量较高的区域却出现了许多缺陷，或者由于开发团队延迟交付软件版本导致压缩测试执行时间等，此时测试经理需要相应变更测试计划，包括测试计划中的风险的变更。测试计划变更通常会要求对后续的测试工作重新进行优先级划分。

对于小型且不太复杂的软件产品，将测试工作产品和测试活动关联到测试目标和测试计划相对容易，同时判断测试是否成功也比较容易。但对于大型且复杂的软件产品，花费一定的工作量进行测试监控框架的定义是必要的。例如，为了决定软件产品版本是否可以及时发布，收集和分析风险、需求、支持的运行环境配置信息，以及测试依据中其他定义的内容是非常有帮助的。

以软件产品利益干系人可理解的方式，将测试工作产品状态、测试活动与测试依据关联起来是非常重要的。测试经理在汇报测试进度、测试结果和测试状态时，就可以按照利益干系人关心的业务全景，清楚地解释测试过程中碰到的问题和困难。

测试计划阶段建立测试依据、测试条件和其他工作产品之间的可追溯性，不仅可以使得它们之间的复杂关系更为透明和容易理解，同时有助于测试经理更好地开展测试监控活动，保持它们之间的可追溯性。测试经理可以基于如下信息与利益干系人进行解释和沟通：哪些质量风险得到了缓解、哪些需求已经实现、哪些运行配置环境已经可以正常工作等。上述信息比纯粹基于缺陷数目、测试用例通过/失败状态等更加容易理解，因为后者的状态信息更偏重测试技术，离利益干系人日常视角过于遥远。

假如软件产品利益干系人要求监督的详细度量项和目标，与软件产品功能或者规格说

明没有直接相关，特别是开发过程中缺少正式文档或只有少量文档时，与利益干系人的协调与沟通就会变得更加重要。例如，采用基于风险的测试策略，识别的质量风险就是沟通的基础；而采用敏捷开发模型的软件项目中，用户故事可以作为测试的基础。测试人员不能错误地认为软件产品没有正式文档输出，就不需要对测试对象进行覆盖率度量。测试必须将度量项与软件产品的质量要求、进度要求等结合进行管理，从而方便利益干系人更好地进行监控。

另外，即使按照软件产品功能定义了规格说明，利益干系人仍可能对软件产品的业务覆盖更感兴趣。因此业务干系人在项目早期的尽早参与，有助于确定测试监控所需的度量项，这不仅可在项目期间帮助提供更好的控制，还可以在整个项目过程中推动和影响测试活动。例如，利益干系人的度量项要求和目标，会影响测试设计和实施的架构，或者测试进度计划，以方便按照这些度量项准确地监督测试进展。

测试经理的职责是确保测试工作的成功。测试经理除了要满足测试策略中定义的测试任务和目标之外，同时还需要满足主要利益干系人的要求。假如测试进度将会影响测试成功（包括无法及时完成测试任务、测试质量低下等），测试经理必须采取合适的控制活动，使测试活动回到正常的轨道上。正如 ISTQB 基础级认证大纲中的描述，控制活动可能属于测试团队的职责范围，也可能需要其他项目利益干系人介入，此时测试经理需要和其他利益干系人协调开展控制活动。测试经理与其他团队构建良好合作关系有助于实施测试控制。

良好的测试计划可以更好地帮助实施测试控制，使测试活动更可视化，也就是说可以更早发现测试活动是否偏离测试计划的要求。同时，良好的测试计划也可以提供更好的度量，从而可以更方便地定位出现偏差的原因，以及采取合适的应对措施解决该问题。测试监控活动必须是基于事实和数据，而非测试经理的主观猜测。

测试控制是一个持续的活动，包括实际进度与测试计划之间的比较，在需要时实施纠正措施。通过测试控制引导测试活动完成测试任务、实现测试策略和满足测试目标，包括在需要时再次进行的测试计划活动。

1.3 测试分析

ISTQB 基础级认证大纲中将测试分析和设计作为一个整体进行描述，而高级大纲将两者分别作为独立的测试阶段进行考虑，以便更好地推动生成测试工作产品，以及更加高效地监控测试过程。测试分析和设计在测试过程中并不完全是串行的，甚至有时存在时间上的并行。因此，测试分析与测试设计之间必定会存在迭代关系：测试设计过程中发现的问题，可以为更深入的测试分析提供反馈。

测试分析阶段的主要任务是确定测试“什么”，ISTQB 术语中定义为“测试条件”，即通过测试分析活动输出一系列的测试条件（也称测试项、测试主题等）。而如何测试测试条件的问题，将由测试设计阶段解决。测试人员既可以分析软件产品提供的测试依据，例如，系统规格说明、设计规格说明、用户故事等识别出测试条件，也可以考虑风险分析过程中识别的产品风险获取测试条件。同时，测试策略中定义的测试目标也是识别测试条

件需要考虑的一个因素。测试分析过程需要考虑不同利益干系人在开发过程中的不同利益和视角。

测试条件可以作为测试监控中的详细度量项的基础，针对测试条件的覆盖率要求也可以作为出口准则的一部分，通过在测试过程中检查测试条件的状态，以评估软件产品是否可以及时发布。例如，出口准则关于测试条件的要求如下：必须达到识别出的测试条件的100%覆盖，并且不能存在与测试条件相关联的严重程度最高的缺陷，即假如有这样的缺陷，至少已经得到修复并经过了确认测试（或再测试）和回归测试。为了达到这个目的，必须保持测试条件、测试用例和测试结果之间的可追溯性。

ISTQB 基础级认证大纲中讨论了典型的测试级别，因此确定测试“什么”必定会随着测试级别的不同而发生变化。尽管不同测试级别的测试分析是相对独立的测试活动，但测试过程中需要对不同测试级别的测试分析结果进行分类和整合，避免遗漏或者重复某些测试任务。当给定测试级别的测试依据已经建立，测试团队就应该尽快进行该级别的测试分析，例如，假如采用的是基于需求的测试策略，则相应的需求文档就是测试分析的主要参考依据；而基于风险的测试策略，识别的产品风险作为测试分析的主要输入。

测试分析可以采用不同测试策略中定义的方法，例如，基于需求和基于风险的集成测试策略方式。测试分析过程中有效识别测试条件，邀请技术干系人和业务干系人的积极参与是非常重要的。

（1）技术干系人，例如，架构师、数据设计师，主要为识别测试条件提供设计规格说明、数据库设计规格说明等技术文档；技术干系人知道潜在的软件失效方式，因此可以从技术层面识别风险和评估其可能性。

（2）业务干系人，例如，业务分析师、产品经理、用户等主要为测试分析提供需求规格说明、用户故事、用例等业务相关的信息。业务干系人了解客户和用户如何使用软件产品，因此可以从业务的角度识别风险和评估其严重程度。

而有的利益干系人既代表业务角度的观点，也代表技术角度的观点。例如，担任测试分析或业务分析角色的专题事务专家，由于其在技术和业务领域的专业知识，通常他们会有更广阔的视角。不管是技术干系人、业务干系人，还是具备两者干系人技能的专家，他们积极参与识别产品风险，并评估不同产品风险的可能性与严重程度，进而得到风险级别，对于成功实施测试分析是不可缺少的。测试分析过程中需要确保测试条件与测试依据之间的可追溯性。

为了通过测试分析得到有效的测试条件，测试经理需要明确回答下面的一些问题：哪些因素会影响测试条件的详细程度？测试条件详细化或者简单化的优缺点是什么？什么场合下适合测试条件详细化或简单化？

1.3.1 影响测试条件详细程度的因素

测试经理在选择测试策略和测试技术时，需要确定以什么样的详细程度描述测试条件（可以理解为测试条件的颗粒度大小）。假如测试条件过于详细，测试人员需要花费相当多的工作量，导致其他测试活动工作量的减少；而测试条件过于简单，可能会导致出现过多的假阳性结果（误报缺陷）和假阴性结果（漏报缺陷）。

尽管测试条件的详细化和简单化之间的平衡很困难，但测试经理还是需要在测试策略中提供一定的指南，指导测试人员以什么样的详细程度描述测试条件，尽量避免出现过度文档化或文档化不足的问题。选择测试条件的详细化程度，会受到各种因素的影响。

1．测试级别

测试条件的详细程度与所处的测试级别有关。典型的 V 模型中有组件测试、集成测试、系统测试和验收测试，其中，组件测试通常有开发人员负责测试。由于时间限制，以及开发人员对作为测试对象的组件非常熟悉，因此很少看见开发人员针对组件（模块或单元等）编写详细的测试条件。但是，假如组件测试中更多采用自动化测试方式，那么就需要考虑测试条件的详细化。一般情况下，集成测试和系统测试需要更正式和详细的文档。

2．测试依据质量和详细程度

技术干系人和业务干系人输出的测试依据，是识别测试条件的基础。测试依据的质量和详细程度会直接影响测试条件详细程度的选择。假如测试依据缺乏详细内容，例如，软件产品需求规格说明，对于采用基于需求的测试策略的测试团队，很难从需求文档中得到详细的测试条件。另外，假如测试依据本身的质量很差（例如，识别的产品风险列表），同样很难指导获取详细的测试条件。

3．软件产品复杂度

软件产品类型和复杂度也是影响测试条件详细程度的一个重要因素。假如软件产品属于安全关键系统，例如，医疗设备系统，测试条件应该是高度详细化的，甚至对软件产品没有深入了解的测试人员，也能理解这些详细的测试条件，判断软件产品的重要内容是否都得到了覆盖。

除了软件产品的类型，其复杂度同样需要考虑。假如软件产品属于大型复杂的产品，测试人员很容易遗漏掉一些测试条件，因此测试团队应该选择详细化测试条件，以确保没有忽略重要的测试范围。

4．项目风险和产品风险

测试过程中识别的产品风险列表，是测试分析的重要参考输入，因此其质量必定会影响测试条件详细程度的选择。除了产品风险，软件开发过程中还会发现项目风险，同样也会影响测试条件详细程度的选择。例如，将软件产品中的部分组件外包给第三方进行测试，即外包测试，假如他们没有对该组件的重要功能进行全面测试，对该项目而言就是一个项目风险。此时如果编写详细的测试条件，作为第三方外包测试必须覆盖的合同的一部分，那么该项目风险就可以得到较好的管理。假如此时提供的是简单的测试条件，发包方和接包方可能会对测试结果和行为出现不一致的解释，从而降低测试的效率和有效性。

5．开发模型与测试过程成熟度

开发过程中采用的开发模型也会影响测试条件详细程度，例如，采用敏捷开发模型，通常会弱化各种文档，其中也包括测试文档；而对于典型 V 模型，前一个阶段输出的文档，是后一个阶段的输入和参考，一般对文档内容和格式要求更高。

测试过程和其他开发过程的成熟度也是影响测试条件详细与否的因素之一，但是该因素不是那么明显。例如，测试过程不成熟，并不能说测试团队必须输出详细文档进行弥补。相对于过程成熟度，软件项目团队的技能，更会影响测试条件详细程度的选择。

6．测试管理工具

测试依据、测试条件和测试用例之间存在紧密的关系，因此跟踪和管理可追溯性的管理工具也会影响测试条件详细程度。例如，测试用例管理工具会定义测试用例的详细程度和格式要求，包括必需和可选的字段等，因此测试条件的选择必定要满足该工具的要求。

7．测试分析人员技能

测试分析人员的技能是影响测试条件详细程度的最重要因素。简单而言，缺乏软件产品相关业务知识和测试技能的测试人员，更倾向详细的测试条件；而熟悉软件产品业务背景且具备深厚测试技能的测试分析人员，简单的测试条件可能是更适合的。

8．其他利益干系人的可用性

开发过程中的其他利益干系人不仅输出软件工作产品作为测试分析的依据，同时，他们也会在测试设计之前参与测试条件的评审工作。因此他们是否有时间参与评审，以及参与后续的测试活动，将直接影响测试条件的选择。例如，在测试分析阶段，测试分析人员可以经常和软件产品的业务干系人进行沟通和讨论，那么测试条件可以相对简单；假如他们只有在测试执行才有时间参与测试活动，那么详细化测试条件是更好的选择。

上面从几个方面简单阐述了可能会影响测试条件详细程度选择的因素，具体选择详细化还是简单化测试条件，测试经理应综合考虑上述因素做出合适的决定。但是测试经理需要牢记的是，测试条件的不同颗粒度一定会影响到测试条件的数目。例如，针对手机闹钟的“铃声”测试，假如“测试音乐铃声”作为颗粒度较大的测试条件，那么在详细化测试条件过程中，该测试条件可以再被分成多个测试条件，例如：

（1）测试不同格式的音乐铃声；

（2）测试存储在不同路径的音乐铃声；

（3）测试不同时长的音乐铃声；

……

1.3.2　测试条件详细化的优点

测试分析阶段选择测试条件的详细化，其主要优点表现在以下几个方面。

首先，测试条件的详细化有助于缺陷预防。创建详细测试条件相比简单测试条件必定需要花费更多的测试工作量，需要测试分析人员更深入地学习和分析与软件产品相关的测试依据。例如，当前处于系统测试的测试分析阶段，测试人员希望从该软件产品的系统需求规格说明中识别详细的测试条件。相比于简单测试条件，为了编写详细测试条件测试人员会更加仔细深入地阅读和学习需求规格说明，而该过程当然可以更多地发现需求文档中的缺陷，从而避免这些缺陷遗漏到后续的开发阶段。因此，测试过程中尽早识别详细的测试条件，可以更好地起到缺陷预防的作用。

更好的缺陷预防，除了体现在软件工作产品，同时也体现在测试工作产品。针对识别的测试条件，测试团队邀请软件产品的利益干系人参与评审。假如测试团队提供的是详细的测试条件，那么利益干系人可以更容易发现其中的遗漏和错误，同时也可以更好地理解测试团队主要关注的覆盖内容。

其次，测试条件的详细化有助于利益干系人更好地理解工作产品。详细的测试条件可

以帮助利益干系人理解测试人员要测试什么，即测试范围。而测试用例的详细步骤通常对于干系人而言过于关注细节，难以理解。另外，以测试条件的方式汇报测试结果，也可以帮助利益干系人理解测试进度状态，这比纯粹的度量数据，例如，已经执行的测试用例数目、失败的测试用例数目等，更容易评估测试进度状态和覆盖率等要求。

第三，测试条件的详细化有助于实现更好的可追溯性。详细的测试条件可以增加与测试依据、测试用例和测试目标的相互关联，可以为特定测试级别内更加清晰的横向可追溯性提供依据。以前面的测试手机闹钟“铃声”为例，假如以“测试音乐铃声”为颗粒度较大的测试条件，那么这个测试条件必然会与许多需求条目进行关联，同时该测试条件也必定与多个测试用例关联，从而导致难以实现有效的可追溯性要求。而详细测试条件则可以比较清晰地显示测试条件与需求、测试用例之间的可追溯性。

第四，测试条件的详细化有助于实现更好的测试过程监控。还是以上面的测试手机闹钟“铃声”测试条件为例，假如以详细测试条件作为测试分析的输出，那么测试设计、实施和执行都会覆盖这些详细测试条件，而不容易遗漏某些测试条件，同时基于测试条件也更容易评估测试覆盖率。基于详细的测试条件，也可以更好地进行测试进度、软件产品质量等的评估和控制。

第五，指导后续测试和开发活动。最后，测试条件的详细化有助于指导后续测试和开发活动。详细的测试条件必定可以更好地进行后续的测试设计、实施和执行。同时详细的测试条件也有助于后续开发活动的开展。例如，详细的测试条件不仅解决了测试人员测试什么的问题，同时也可以让开发人员知道测试人员准备如何检查他们的工作产品，因此开发人员可以基于测试条件检查和更新他们的需求规格说明和设计规格说明等软件工作产品。

1.3.3　测试条件详细化的缺点

测试条件的详细化具备上述的一些优点，也存在如下一些缺点。

首先，相对于测试条件简单化，测试条件详细化必定会导致测试人员花费更多的时间和成本，即便是小型化的软件项目。而测试工作量相对是有限的，测试条件的识别和细化花费更多的工作量，必定会导致其他测试活动工作量的压缩。

其次，测试条件详细化除了前期的费时费力之外，维护详细化的测试条件不仅工作量巨大，对测试团队而言也是一个挑战。特别是针对采用迭代-增量开发模型的软件产品，因为不断有新的功能增加或原有功能的更新，这都会导致对原来的测试条件进行更新，即测试条件难以维护。

第三，假如采用测试条件详细化的原则，则需要在整个测试团队内部定义一致的详细程度和具体格式等要求。也需要在测试团队内部进行培训，以帮助他们在创建测试条件和维护文档方面达成一致。测试过程中经常存在一个问题：尽管测试团队内部定义了统一的测试条件详细程度要求和模板，但是在实施过程中经常出现要求和模板流于形式的问题。

1.3.4　测试条件详细化的适合场景

测试条件详细化有其优点和缺点，测试经理在选择具体详细程度时必须要综合考虑上

述的各种影响因素。选择详细的测试条件，在如下场景下会更适合。

首先，测试团队计划在测试设计和实施过程中采用轻量级测试文档，那么测试条件详细化是比较适合的。例如，测试过程中为测试分析阶段预留了较多时间和资源，而在测试设计和测试实施等阶段预留的时间较少，此时测试分析阶段获取详细测试条件是合适的。又比如，当前软件项目采用的是敏捷开发模型，详细的验收标准是必需的，同时又尽量避免重量级的测试文档输出。

其次，假如开发过程中输出的测试依据不全甚至没有，则建议采用详细的测试条件。当测试团队无法根据测试依据创建测试条件时，测试人员必须与软件产品利益干系人紧密合作，例如，通过业务干系人和技术干系人获取详细的软件产品的需求规格说明和设计规格说明等。测试人员在测试分析、设计和实施阶段，与不同利益干系人的沟通与交流，不仅可以帮助获得详细测试条件，同时也可以作为反馈弥补测试依据的不足，从而更好地指导开发人员进行软件产品的分析、设计和实现。

第三，假如软件产品属于规模大且复杂的高风险类型，例如，安全关键系统，此时采用详细测试条件也是合适的。复杂程度高且规模大的软件项目，测试团队面对的必然是成千上万的测试用例，测试用例与其他开发工作产品之间的可追溯性和可维护性都是十分困难的。另外，对于一些高风险的软件产品，满足特定的法律法规是强制性要求，必须有明确的测试用例覆盖所有的软件产品需求，并能满足外部审计的要求。同时，详细测试用例也有助于高风险软件产品的监督和控制，而通过简单的测试用例与开发工作产品关联是很难实现有效监控的。

1.3.5 测试条件简单化的适合场景

上面讲述了测试条件详细化的优点、缺点和适合的使用场景，而测试条件简单化可以认为与详细化具有相反的特点。以下是适合于测试条件简单化的使用场景。

首先，假如设计和实施的测试用例与测试依据之间有清晰的对应关系，例如，比较简单的软件产品，其测试什么和如何测试之间存在简单的层次关系，那么采用测试条件简单化的方式是比较适合的。测试用例和测试依据的简单映射关系，可以表现为以下三层意思。

（1）测试依据中获取测试条件简单明了；

（2）测试依据中提供的测试条件通过/失败准则简单明了；

（3）测试依据中需要测试验证的条目与测试条件属于简单的对应关系，例如，1 对 1 或者 1 对 2 之类的对应关系。也就是分层次结构清晰。

其次，测试条件简单化的常见的一个使用场景是组件测试。相对而言，每个组件涉及需要实现的功能比较简单，例如，类或者函数等，因此不需要识别很多的测试条件进行覆盖，即测试条件简单化是合适的。

第三，测试条件简单化的另一个使用场景是验收测试，其前提条件是针对该软件产品定义了丰富的用例（Use Case）。测试条件的详细内容已经通过用例的方式描述清晰。

☆示例：测试分析阶段识别测试条件

针对 iBAS R1.0 项目的 IGMP 功能测试。IGMP 功能子模块包括：协议处理模块、组播控制模块、用户管理模块和组播流管理模块。

测试团队针对 IGMP 功能的测试，采用基于风险的测试策略，识别的产品风险作为开展后续测试活动的基础，例如，基于产品风险设计测试用例、安排测试优先级、分配测试资源等，并通过执行测试用例缓解产品风险。IGMP 功能相关的技术干系人和业务干系人积极参与产品风险的识别、评估和缓解等活动对于成功实施基于风险的测试策略是至关重要的。保持测试用例和产品风险之间的可追溯性也是测试经理需要关注的重点。可以通过下面几个方面检查可追溯性的完整性。

（1）可以根据 ISO 9126 质量模型提供的 6 大质量特性，例如，功能性、可靠性、效率、易用性、可维护性和可移植性分类产品风险。假如测试分析阶段识别的产品风险，没有包含上述某些质量特性，测试团队需要检查在风险识别过程中是否遗漏了某些产品风险。假如是，那需要通过业务干系人和技术干系人的支持识别更多的产品风险，并将它们添加到上述产品风险列表。

（2）将 IGMP 功能需求规格说明中的需求与产品风险列表相关联，确保每条需求至少有一个产品风险与之对应，有些需求与产品风险之间也可以存在多对多的对应关系。假如某些需求没有与之关联的产品风险，测试团队需要识别新的产品风险与之对应，并在需求和产品风险之间构建可追溯性。

（3）在测试执行阶段，测试团队还需要将发现的缺陷与产品风险进行关联。假如测试执行中采用了应对型的测试策略，例如，探索性测试，此时将缺陷与产品风险的关联将显得更加重要。假如缺陷并不是通过已经识别的产品风险发现的，那么需要分析什么产品风险可以发现该缺陷，并将此产品风险添加到产品风险列表，并以测试用例的方式对产品风险进行说明。

需求或者产品风险与测试用例和缺陷之间的可追溯性要求可以通过测试管理工具或者其他数据库工具实现。

1.4 测试设计

测试分析阶段识别的测试条件，确定了测试“什么”，而测试设计阶段是确定“如何”测试的活动。测试设计通过使用测试策略或者测试计划中定义的测试设计技术，有效地将识别的测试条件转化为测试用例。

iBAS R1.0 项目的 IGMP 功能测试（参考 1.3.5 节的描述），测试过程中以图形化界面作为测试人员测试的配置参数的界面，假如测试团队针对该界面采用基于需求的测试策略，需求中定义了该图形化用户界面的各个参数，以及每个参数的输入要求。假如测试分析师定义了如下的测试条件：检查图形化用户界面，针对无效输入，系统是否会拒绝接受输入并给出相应的错误信息？

首先，测试分析师根据提供的需求规格说明获取图形化用户界面每个输入参数的有效输入值的定义。通过分析以前软件产品所发现的缺陷列表后，测试分析师发现其中的很多问题都与输入无效值有关，即在处理无效输入时经常出错。因此，测试分析师决定采用等价类划分和边界值分析，以获取无效的等价类和边界值，并作为设计测试用例的输入值。

其次，确定测试用例的输入之后，还需要确定每个输入之后得到的期望结果，即 IGMP

功能处理需要输出的错误信息。同样，IGMP 功能需求规格说明应该会提供，针对什么样的异常输入，输出什么样的错误信息，或者从业务干系人或者技术干系人得到期望结果。测试分析师将输入、操作步骤和期望输出结果进行文档化。

第三，根据测试计划或者测试策略中定义的测试设计技术确定测试输入之后，测试团队面临的问题是生成概要测试用例（逻辑测试用例，详细内容参考 1.4.2 节）还是详细测试用例（实际测试用例，详细内容参考 1.4.2 节），测试分析师根据确定的详细程序开发测试用例。作为测试用例设计的重要组成部分，测试分析师必须确定测试用例运行所需的测试数据和测试环境。

假如能邀请业务干系人和技术干系人对测试用例、测试数据和测试环境进行评审，不仅有助于尽早发现测试用例中的遗漏和问题，同时可以让干系人了解测试的关注点，并更好地获得他们的支持。完成测试用例、测试数据和测试环境之后，测试经理还需检查测试设计的出口准则。假如出口准则的要求都满足了，那么可以开始测试实施阶段。

1.4.1 测试设计中的可追溯性

测试设计过程中，测试人员创建、维护和更新可追溯性信息是非常重要的。不同的软件开发生命周期、测试过程和测试团队，其采用的可追溯性方式是不一样的。例如，可以将测试用例直接与测试依据中的内容相关联；也可以通过测试条件将测试用例与测试依据之间进行关联。测试团队在时间和资源允许的情况下，更建议通过测试条件将测试用例与测试依据维持可追溯性，因为通过测试条件保持可追溯性可以更好地对测试质量、进度等进行监控。测试设计中实施可追溯性，需要注意两点：可追溯性的粒度和可追溯性的双向要求。

首先，保持可追溯性的粒度要求。以系统需求规格说明为测试依据作为例子，测试人员在测试设计时，有时会将多个需求条目映射到一个测试条件，或者一个测试条件设计了多个测试用例进行覆盖。此时，假如测试执行过程中，其中的一个测试用例由于发现了缺陷而处于“失败”状态，就会出现该测试条件是否处于“失败”状态的疑问。或者需要判断到底是哪个需求没有得到满足。这对测试人员是一个挑战，同时也对测试管理带来一定的影响。因此，为了在测试过程中对出现的这些情况进行更好的应对，测试经理应该在测试计划中，明确说明需求、测试条件和测试用例之间存在多对多的映射关系，如何定义其中的“失败”和“通过”准则。

其次，保持可追溯性的双向要求。可以是从需求、产品风险，或者其他测试依据内容到测试条件，再到测试用例，最后到测试执行过程中的测试结果，以及缺陷报告之间保持可追溯性，简称为从前往后的可追溯性。或者根据测试执行的测试结果和发现的缺陷，再回溯到测试用例、测试条件，再到需求、产品风险等，简称为从后往前的可追溯性。作为测试经理，测试过程中保持良好的可追溯性，不仅可以更好地实施测试过程监控，同时可以与利益干系人更好地进行沟通，例如，告诉利益干系人成功运行了多少个测试用例，失败多少，发现多少缺陷等数据，其直观意义不如告诉利益干系人，哪些需求没有实现或者哪些产品风险没有得到缓解。

另外，测试过程中保持可追溯性的另一个重要意义是：有助于进行变更影响分析。对

测试团队而言，由于维护了测试工作产品与开发工作产品之间的可追溯性，假如相应的开发工作产品出现变更，就可以更好地进行变更的影响分析，测试经理也可以基于影响分析的结果，更新测试计划，例如，更新测试估算结果、调整测试优先级、回归测试的深度和广度等。另外，可追溯性也可以帮助开发团队在变更时进行整个软件产品的影响分析，以更有效地开展和监控后续的活动。

1.4.2　概要测试用例和详细测试用例

测试条件是测试设计的输入，但测试条件只是告诉测试人员测试“什么”，并没有说“如何”测试。因此，测试分析人员在测试设计过程中，必须明确测试条件相关参数的输入域，针对每个无效输入的错误信息等。根据测试类型和选择的测试技术的不同，测试分析人员所需的软件产品信息是不同的。另外，测试用例中包含信息的详细程度，决定了测试人员能获取的软件产品信息详细程度：详细测试用例需要更多的产品信息，而概要测试用例所需的信息相对较少。

（1）概要测试用例：或称为逻辑测试用例，指的是没有定义输入数据值和期望结果的测试用例。例如，只是定义了操作符，但实际的输入值并没有定义。

（2）详细测试用例：或称为实际测试用例，指的是定义了具体的输入数据值和期望结果的测试用例。例如，将概要测试用例中所使用的逻辑运算符替换为对应于该逻辑运算符作用的实际值。

还是以 1.3.5 节的 IGMP 功能的测试条件“检查图形化用户界面，针对无效输入时系统是否会拒绝接受输入并给出相应的错误信息”为例，说明概要测试用例和详细测试用例之间的区别。概要测试用例描述了测试分析师采用的测试设计技术，例如，等价类划分和边界值分析，以识别输入的无效等价类和边界值，同时也描述了去哪里查找测试用例的期望结果，这里指的是错误信息。在概要测试用例中不会包含具体的错误信息。而详细测试用例步骤中包含所有实际的非法输入数据，以及期望输出的特定错误信息。

开始测试设计之前，测试分析师必须明确具体采用的是概要测试用例还是详细测试用例，否则将会导致后续问题的发生。例如，假如测试过程只需概要测试用例，而测试分析师设计了详细测试用例，导致更多成本用于详细测试用例的维护和更新。详细测试用例还是概要测试用例，测试经理应该在测试计划中明确定义。

测试实践过程中，详细测试用例还是概要测试用例并不是那么的绝对，更多的是两者的折中和平衡。不同的测试人员会设计完全不同详细程度的测试用例，因此测试经理需要在测试计划中定义其中的平衡要求和手段。

另外，测试用例设计是一个不断迭代和详细化的过程。通常，首先定义概要测试用例，后续再不断细化得到详细测试用例。假如采用应对型测试策略，例如，探索性测试，可以将每个会话中的测试章程看成是概要测试用例。在测试执行阶段再将测试章程转换为详细测试用例，尽管这个过程不一定是文档化的。需要注意的是，每个概要测试用例都需要转化为详细测试用例（但不一定文档化），因为在测试执行过程中必定需要提供输入数据，并根据期望结果与测试得到的实际结果进行比较，假如两者出现不一致的情况，在确定是被测对象原因时提交缺陷报告。

1.5 测试实施

测试实施阶段的主要任务包括：概要测试用例转换为详细测试用例、设置测试执行优先级、准备测试数据、测试用例自动化、检查测试执行入口准则，以及确定测试执行进度。

假如测试过程输出的文档体系遵循 IEEE Std 829—2008 标准，测试分析阶段输出测试设计规格说明（包含测试条件），测试设计阶段输出测试用例规格说明（包含概要测试用例和详细测试用例），而测试实施阶段输出测试规程规格说明，得到包含具体数据和期望结果的测试用例，并通过不断迭代的方式最终得到测试规程，其中定义了不同测试用例建议的执行先后顺序。当然，很多测试团队并不会完全遵循 IEEE Std 829—2008 标准的文档体系，例如，针对测试用例只输出一篇文档，其中包含输入、期望结果、步骤，以及测试用例执行的优先级等。

在测试设计阶段对测试数据的范围和要求进行了整体设计，而在测试实施阶段需要对数据进行实施，例如，测试数据的匿名化、大容量数据的准备、特殊要求的数据准备等。有的情况下，测试团队会获取软件产品的实际使用的数据作为测试数据，例如，通信产品中的用户认证信息，此时测试团队必须对测试数据进行安全性处理，如测试数据的清洗或者匿名化，以去除任何个人隐私信息，同时保证数据的内部完整性要求。处理后的数据可以用于测试执行中的输入，以避免安全漏洞或者误用个人隐私信息带来的风险。这对于需要大规模的真实数据的软件产品测试是非常重要的。

测试数据准备之后，需要将它们导入测试环境，因此测试实施阶段完成测试环境准备工作也是必需的。不管是手工测试还是自动化测试，测试环境都需要根据测试设计中的要求进行。测试环境的准备可以在手工测试或者自动化测试实施之前开始，或者说与测试实施并行进行。特别是自动化测试，测试环境的尽早准备尤为重要，因为自动化测试的开发和调试都依赖于测试环境的实施。另外，通过早期的测试也可以确定测试环境是否得到了正确的配置，这也是测试实施阶段的重要工作。测试环境准备工作通常需要其他团队的支持，例如，内部网络团队，这需要测试经理在测试计划阶段为其预留工作量，并与其他支持团队进行沟通与协调。

当测试用例、测试数据和测试环境准备好之后，测试经理开始检查测试执行的入口准则是否已经满足。测试执行的入口准则应该在测试计划中明确得到定义。测试执行入口准则不仅包含对测试团队的要求，同时对其他项目团队也有要求，例如，开发团队是否及时交付了被测的软件版本等。

当测试执行入口准则满足之后，测试过程就进入测试执行阶段。测试执行入口准则是否满足并不是测试经理单方面的判断，通常是管理团队共同评审的结果，包括业务干系人和技术干系人的参与。入口准则评审中，测试经理不仅需要获得可以开始测试执行的决定，而且需要得到利益干系人对测试执行提供支持的承诺，例如，支持测试环境和测试数据的更新和维护。

1.5.1 测试执行优先级

不管是以概要测试用例还是详细测试用例，或者以测试脚本的形式对测试用例进行文

档化，测试实施阶段都需要对它们进行优先级设置。很多情况下，测试团队可以把测试用例组织成测试套件（例如，测试用例组）是有意义的，因为这样有助于更好地组织测试执行活动，相关的测试用例可以一起执行。测试执行优先级的设置会受到各种因素的影响，因此测试团队在考虑优先级时需综合考虑。

首先，测试策略会影响测试优先级的设置。例如，采用基于风险的测试策略，那么测试优先级的确定可以来源于测试用例的风险级别。根据不同的测试覆盖率要求和测试团队经验等因素，测试执行的先后顺序可以采用基于风险的深度优先策略或者广度优先策略进行实施（深度优先与广度优先的内容，请参考 2.3 节）。假如测试团队采用基于需求的测试策略，那么测试依据，例如，需求规格说明中的需求条目的优先级，可以帮助设置测试用例的优先级。

其次，测试资源是否及时到位也会影响测试优先级的设置。例如，负责某些关键产品功能的核心测试人员具体到位的时间，将确定该功能模块在什么阶段进行测试。测试设备、测试数据等的到位时间同样会影响优先级的安排。

第三，开发团队相关的因素也会影响测试优先级的设置。例如，开发团队提交软件测试版本的进度安排。因此，测试团队必须与开发团队就被测软件的版本交付进行紧密沟通和协调，特别是在采用迭代-增量开发模型中，非常重要的一点是测试经理通过和开发团队沟通以确认软件版本按照可测试的顺序交付。

测试实施阶段考虑手工测试和自动化测试的运行顺序是非常重要的一项任务，同时需认真检查需要按照特别顺序运行的各种限制因素。假如可能，将它们之间的依赖关系文档化。同时，测试执行优先级并不是一成不变的，测试执行过程中，测试经理需要根据测试资源、发现的缺陷、开发团队的变更等因素，不断动态调整测试执行优先级，从而提高测试效率和有效性。

1.5.2　测试执行入口准则

为了保证测试执行的顺利进行，测试经理需要在测试实施阶段检查测试执行工作是否可以正式开始，即是否满足测试执行的入口准则。良好的测试执行入口准则不仅有助于测试活动的顺利进行，同时也可以帮助测试人员在测试过程中利用这些准则评估和判断测试状态，并根据不同的测试状态采取合适的措施。

作为测试经理，需要在测试计划阶段认真确定哪些条件是必须在测试执行之前满足的。假如前期的入口准则条件没有得到很好的满足，测试团队可能会在后续碰到一些不可预料的事件。下面是测试执行入口准则需要考虑的一些条目。

首先，测试用例相关的文档已经得到评审和更新。假如测试过程遵循 IEEE Std 829—2008 测试文档标准，测试经理需要确保测试执行之前，被测对象的测试设计规格说明、测试用例规格说明和测试规程规格说明都已经准备就绪，经过业务干系人和技术干系人的评审，并根据评审建议进行了更新。假如测试过程中还实施测试执行的自动化，那么相关的自动化测试用例和测试脚本同样也需要得到评审和更新。

其次，测试环境准备就绪，包括测试人员、测试工具、测试设备、测试数据等。测试环境是否准备就绪直接影响测试执行工作的顺利开展。例如，在测试 iBAS IGMP 功能过程

中，相关的测试设备没有及时到位导致整个测试进度出现延期。而其没有到位的原因是由于海关检查时缺少一个文件，而导致无法入关。因此，测试经理应对关键的测试环境设备和工具尽早进行风险识别、评估和应对，并尽早制定应急计划。测试执行开始前检查测试数据的完备性和正确性，也是测试环境检查中的一环。

第三，检查被测软件版本的最基本质量（冒烟测试）。尽管测试经理无法控制开发提交的软件版本的质量，但是测试经理需要确保被测软件版本可以正常运行，同时要求开发团队提交软件版本的同时，必须提交该软件版本的说明，包括该版本中新增加的功能特性、修改的缺陷、没有修改的缺陷、可能存在的问题以及测试重点的建议等。

前面罗列了测试执行入口准则的常见的几个条件选项，测试经理需要根据开发模型、测试过程和项目团队的不同特点进行合理的裁剪。而验证这些测试执行入口准则是否满足条件的一个好的实践是：测试执行之前开展冒烟测试。基于前面的测试环境、测试软件版本和测试数据，选择部分测试用例构建一个测试集（例如，针对被测对象核心功能的一些基本测试用例），只有全部执行通过，才可以认为是满足测试执行的入口准则。至于是由开发团队还是测试团队来执行冒烟测试，可以在前期项目计划中进行定义。另外，由于会经常进行冒烟测试，因此将冒烟测试相关的测试用例进行自动化是一个不错的选择。

下面是 iBAS R1.0 IGMP 系统测试计划中定义的系统测试执行入口准则。

（1）IGMP 系统测试设计规格说明、IGMP 系统测试用例规格说明、IGMP 系统测试规程规格说明和 IGMP 系统测试自动化用例经过评审并且通过；

（2）IGMP 自动化测试用例在实际的测试环境中验证通过；

（3）IGMP 的集成测试报告已经提交并评审通过；

（4）IGMP 测试所需的测试资源，如测试仪表、测试平台、测试人员等已经到位；

（5）开发人员提交了 IGMP 的版本说明，其中阐述了从开发角度建议的测试重点、测试建议、存在的缺陷和问题列表等；

（6）提交的 IGMP 软件版本在测试环境中通过了冒烟测试。

1.5.3 测试执行进度

制定测试执行进度是测试实施阶段的另一个重要测试活动，测试执行进度并不是一成不变的，测试经理应该随着软件项目的深入或者测试活动的持续开展不断进行更新。测试执行进度制定过程中，测试经理需要考虑下面的一些因素。

测试执行进度主要依据测试用例优先级来制定。测试用例优先级可以依赖于对应需求的优先级、产品风险的级别、被测对象的相互依赖关系、测试资源的限制等方面的因素。详细内容，可以参考 1.5.1 节。

测试执行进度的制定会受到各种测试限制条件的约束，从而影响测试用例的高效运行。例如，具备特定测试技能的测试人员是否及时到位；更改某个特定配置的测试环境之前，是否能够将与之相关的测试用例全部高效运行完成，因为测试环境的修改和配置需要花费大量的工作量；测试用例之间的依赖关系，开发团队发布被测对象功能模块的先后顺序等，都是影响测试用例高效运行的因素。

测试执行进度应该同时考虑手工测试和自动化测试。假如测试资源和成本允许，建议

将手工测试和自动化测试环境分开，以避免两者测试的相互影响。假如在同一个测试环境开展自动和手工测试，可能会导致测试时间的浪费。自动化测试的一个优点是可以利用周末和晚上时间，但假如自动化团队没有和手工测试团队很好地协调测试环境，例如，自动化测试将前面手工测试配置的测试数据进行修改或者重置，就会导致测试结果的混乱，甚至出现误报缺陷（假阳性结果）的情况。

测试经理制定测试执行进度时，应该邀请测试团队成员参加，并对执行进度进行仔细评审，综合评审测试用例重要性、优先级、测试资源限制、依赖关系等，以帮助测试团队可以更高效地完成测试用例的执行。图 1-1 是针对 IGMP 系统测试制定的测试时间进度的部分内容。其中包含测试执行进度的甘特图。

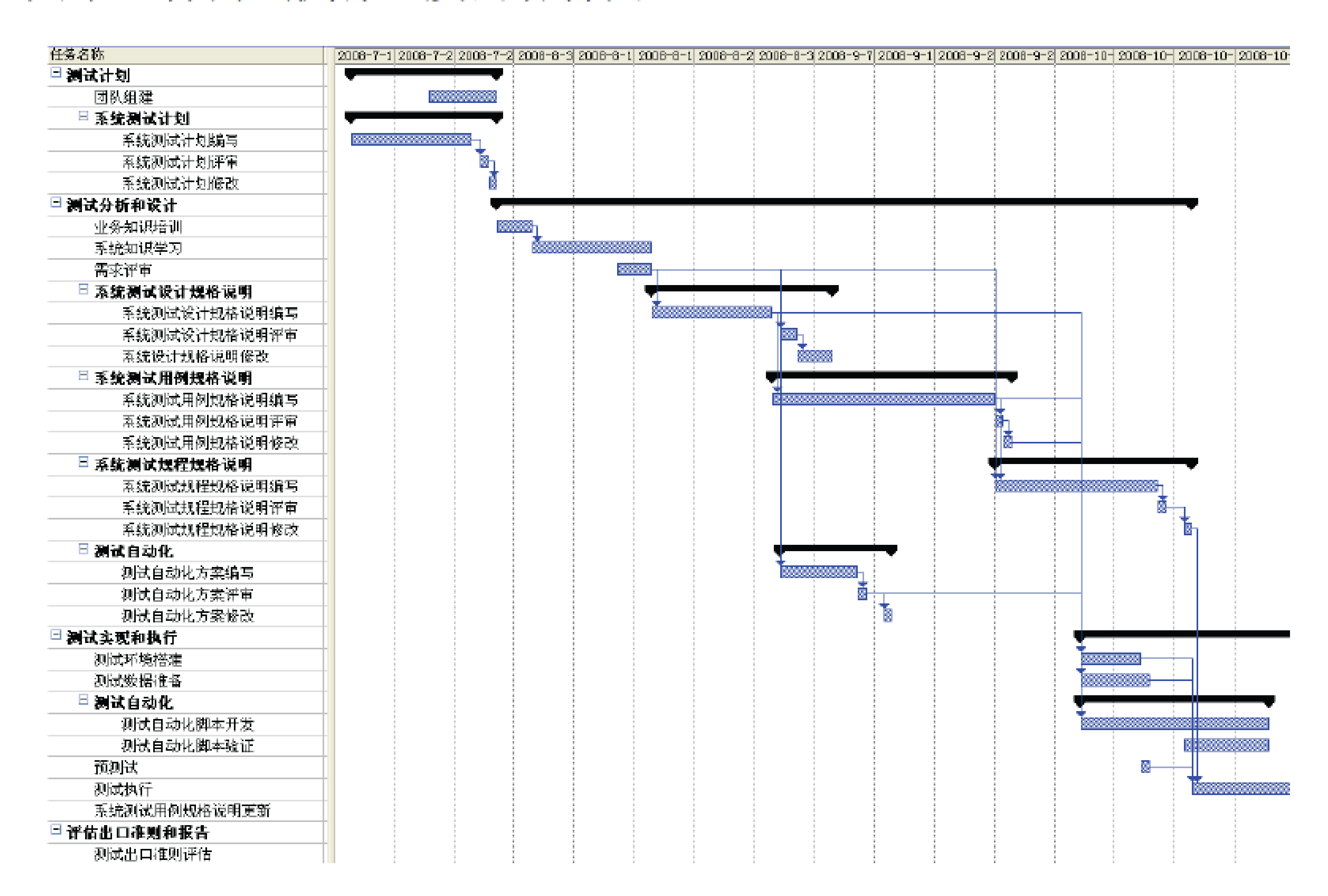


图 1-1　IGMP 系统测试进度甘特图（部分）

1.5.4　尽早测试实施的优缺点

测试实施并不是等完成测试设计之后再开始的，假如时间和资源允许，测试经理应该让相关测试人员尽早参与测试实施活动。测试团队尽早开展测试实施活动，其优点和缺点各是什么？

优点 1：测试团队尽早创建详细测试用例，有助于获得被测对象的详细信息，而这些信息可以是与软件产品的业务干系人、设计人员和编码人员等进行讨论而获得。通过测试用例的方式，业务干系人可以发现验证测试用例比验证抽象的业务相对容易，并进一步可以发现规格说明中的缺陷，以及其中的一些模糊需求，起到缺陷预防的作用。测试用例的不断优化，同时也可以为开发团队进行设计和开发工作提供启发式的说明。

优点 2：尽早测试实施也有助于避免测试执行的延期。测试环境准备和验证，测试执行入口准则验证等，也是测试实施过程的重要活动。测试过程中有时会发生这样的情况：尽管测试执行的入口准则没有满足，但是管理团队基于市场方面的压力，允许开发团队提交软件版本，要求测试团队提前开始测试执行活动。假如测试团队能尽早做好测试环境、

测试数据等准备工作，开发团队尽早提交可测试的版本，就能尽早发现测试执行时的问题，保障测试执行的顺利展开。当然，在没有满足测试执行入口准则情况下进行测试执行工作，可能会带来其他的项目风险，这是测试经理需要注意的地方。

尽早测试实施可以带来一些优点，同时也会存在一些缺点。尽早准备测试用例、测试数据和测试环境，有时候并不是合适的策略。假如被测对象的需求规格说明和数据结构发生了变化，必然导致相关测试工作产品的不断变更。这类似于测试团队一直追着一个不断变化的目标在开枪。特别在采用敏捷开发模型的软件项目过程中，因为要“拥抱变化”，每个迭代中都会碰到不断的变更。敏捷开发中每次迭代代码发生巨大的变化，都会造成大量的测试实施工作被浪费。任何增量或者迭代软件开发生命周期都可能发生变更，使得脚本化测试变得不可靠和需要更高的维护成本。即使是顺序型软件开发生命周期，如果项目管理不当，需求发生频繁变更，甚至在项目的后期出现需求变更，其导致的相应变更将会极大地增加成本和浪费时间。因此，测试经理在开展尽早测试实施之前，了解软件开发生命周期和软件功能的可测试性是明智的。

假如测试团队已经花费大量的工作量在实施详细测试用例、测试环境和测试数据，那么上述的变更必定会导致大量的返工。更糟糕的情况是，变更可能会导致前期的测试工作成为无用功，测试团队不得不重新构建新的测试工作产品，这将会带来更多的维护工作量，同时也可能在不同的软件工作产品中引入缺陷，导致产品质量的下降。

尽早进行测试实施活动，既有优点也有缺点，如何选择将是测试经理面临的一个挑战。测试经理必须考虑什么时候开始测试实施，以及以什么样的详细程度生成测试工作产品。测试经理需要在测试工作产品的创建工作量、维护工作量、缺陷预防等方面进行平衡。

1.6 测试执行

一旦满足测试执行的入口准则，测试团队就可以开始测试执行，例如，测试环境已经搭建完毕并进行了验证、开发团队提交了软件测试版本等。测试环境的搭建和验证，通常需要网络运行团队的支持（负责构建测试实验室的人员，不同组织其称呼不同）；而软件测试版本通常需要配置管理团队的支持。假如入口准则的某些要求没有得到满足，测试经理需要分析和评估由此引起的风险和额外工作量。另外，测试过程需要使用各种测试工具，特别是用于测试管理、缺陷跟踪和测试自动化执行的工具；测试团队成员需要为测试结果的跟踪提供各种度量数据，同时所有团队成员都应该理解跟踪的数据。测试经理要提供并公布测试日志和缺陷报告的标准模板；确保在测试执行之前就完成这些事项，从而可以高效地开展测试执行活动。

当测试团队安装第一个软件测试版本之后，测试分析师和技术测试分析师就可以开始执行测试用例。测试分析师主要关注手工测试用例的运行，假如采用应对型测试策略，探索性测试和其他基于经验的测试技术也是测试分析师的关注点。自动化测试用例的执行，可以是技术测试分析师的职责，也可以在技术测试分析师支持下由测试分析师具体开展。

测试执行的结果需要记录在测试日志中。假如测试执行得到的实际结果和测试用例的期望结果有差距，测试分析师需要确认是一个误报问题（假阳性结果）还是一个缺陷。假

如通过重现发现是由于被测对象原因引起的失效，测试分析师需要在缺陷管理系统中提交缺陷报告；假如属于假阳性测试结果，测试分析师需要确认是测试用例、测试数据还是测试环境导致的，对它们有针对性地进行更改，并再次运行相关的测试用例。

为了不断评估测试执行中发现的缺陷，定期的缺陷评审会议是必需的。业务干系人和技术干系人通常都需要参与缺陷评审会议，其主要目的是确认提交的问题是否是缺陷、缺陷严重程度、缺陷优先级、计划在哪个软件版本修复等。关于缺陷管理方面的内容，可以参考第 4 章。

业务干系人和技术干系人组成的团队，需要根据测试日志提供的信息，评估测试进度和被测产品的质量。假如当前的测试状态和质量情况与测试计划要求之间出现超出阈值的偏差，管理团队需要采取合适的控制手段，使其逐步恢复到测试计划目标上。

最后，随着测试执行的不断进行，测试进度和状态的评估会议的频率会逐步增加。在测试执行后期，由业务干系人和技术干系人参与的团队需要对是否满足出口准则进行评估，以确定测试是否可以顺利结束。作为测试经理，必须准备评估出口准则所需的各种度量数据，帮助不同的利益干系人更好地评估测试进度和产品质量。

1.6.1　选择合适的测试策略

测试策略描述了组织为达到预定的测试目标而采用的测试方法，包括产品风险和项目风险管理、测试级别和阶段定义，以及测试的概要活动等。测试策略可以基于测试活动开始的时间而进行分类：预防型策略（尽早设计测试用例以预防缺陷）和应对型策略（测试活动在被测对象实现之后才开始）。典型的测试策略（详细的关于测试策略的内容，请参考 2.4.2 节）如下。

（1）分析型的测试策略，例如，基于风险的测试。

（2）基于模型的测试策略，例如，基于运行概况的测试。

（3）基于方法的测试策略，例如，基于质量特性的测试。

（4）符合过程或标准的测试策略，例如，基于 IEEE Std 829—2008 的测试。

（5）应对型的测试策略，例如，基于已知缺陷的攻击测试。

（6）反回归测试的测试策略，例如，采用广泛的自动化进行回归测试。

上述典型的测试策略并不是孤立存在的，测试过程中测试经理应该根据实际情况，组合应用不同的测试策略，即采用混合的测试策略，例如，在基于质量特性的基础上，采用基于风险的测试。

应对型的测试策略通常是混合测试策略的组成部分，例如，探索性测试，在测试过程中包含 20%～30%的测试工作量用于应对型的测试活动。实践证明，探索性测试可以很好地弥补预防型测试策略中存在的不足，例如：提供更好的发散性测试思维、有效解决测试过程中的测试用例杀虫剂效应。测试经理应该在测试过程中鼓励采用不同的测试策略，因为它们本身具备了不同的优点和缺点，从而可以达到较好的互补功能。

如何将应对型测试策略，例如探索性测试融合到测试过程中？下面的几个建议可以作为参考。

首先，测试经理可以鼓励测试人员在测试执行时，超越书面规定的测试用例输入、步

骤以及它们的组合。即测试人员以测试用例为主线，但可以在执行过程中增加或者变更一些备选的路径。这就像旅游，尽管你的旅游线路是到达某个旅游景点，但假如路途中有吸引你的地方，可以停下来并享受其中的风景。例如，计划一个小时的测试用例执行，其中预留 10～15 分钟用于探索性测试活动，测试经理不仅可以很好地基于测试过程监控测试进度，同时也可以发挥探索性测试的优点。

其次，测试经理可以鼓励更有经验的测试人员开展更多的应对型的测试活动，例如，采用 ISTQB 基础级大纲中定义的基于经验的测试和基于缺陷的技术。通过采用测试章程和测试会话的方式，测试经理也可以高效地管理测试过程。

第三，相比预防型的脚本化测试，应对型的测试在缺陷管理方面更加困难。假如开发人员质疑测试人员的测试结果，测试人员在收集测试的输入、步骤、测试数据和测试环境等方面会面临挑战，因此也更难回答开发人员的疑问。另外，应对型的测试在测试对象和内容方面也比较发散，经常出现测试活动和会话章程不一致的情况。Jonathan Bach 提出的“基于会话的测试管理”（Session-Based Test Management，SBTM）方法[①]，可以更好地管理探索性测试活动，同时更好地发挥探索性测试的优势。

1.6.2 测试经理在测试执行中的职责

尽管测试经理不参与具体的测试用例执行，但其在测试执行过程中的缺陷管理和测试状态监控等方面却发挥着重要的作用。

作为测试经理，测试执行阶段应该每天深入测试团队以了解测试状态，其中每天召开简短的测试进度会议是一个不错的选项。通过每天简短的测试会议，测试经理可以与测试人员讨论运行了哪些测试用例，执行通过的和失败的测试用例有多少，是否存在被阻塞的测试用例以及被阻塞的原因，提交了多少缺陷报告，严重的缺陷是什么等。测试经理得到这些测试信息之后，不仅可以了解测试状态，同时可以有针对性地对测试进度和缺陷管理状态进行更新。通过与测试计划中定义的测试目标、测试策略和进度计划进行比较，测试经理可以确定当前测试进度是否处于可控状态。

获取当前测试状态，是测试经理针对测试进度开展有效测试控制的前提。根据 ISTQB 高级大纲的要求，测试进度监控可以从下面 5 个方面开展。测试进度监控的详细内容，请参考 2.6 节。

（1）产品风险。例如，测试用例执行通过而得到缓解的产品风险百分比。

（2）缺陷。例如，缺陷在被测对象中不同功能模块的分布。

（3）测试（或测试用例）。例如，已计划的、已实施的、已运行、通过的、失败的、无法执行的和跳过不执行的测试总数。

（4）覆盖率。例如，需求和所设计功能的覆盖率。

（5）信心。例如，测试人员对被测对象质量的主观评估。

假如当前测试进度状态与测试计划定义的目标出现偏离（超出偏离的阈值范围），测

① Jonathan Bach. Session-Based Test Management. Software Testing and Quality Engineering Magazine, 11/00.

试经理必须采取合适的应对措施进行控制，使得测试进度回到测试计划的目标上。根据测试过程中获取的测试状态和信息，测试经理可以采用不同的策略和方案对测试进度进行控制，例如：

（1）更新产品风险、测试优先级或测试计划；

（2）增加资源或增加测试的工作量；

（3）推迟软件版本的发布时间；

（4）降低或加强测试出口准则。

测试过程中保持可追溯性，不仅可以进行测试覆盖率的评估和变更影响分析，同时对测试进度的监控也有重要作用。测试过程中的可追溯性，可以体现为系统需求、测试条件和测试用例之间可追溯性，也可以是产品风险、测试条件和测试用例之间的可追溯性等。以测试进度监控 5 个维度中的产品风险为例，在测试过程的早期通过头脑风暴形式，尽量多地识别产品风险，并在测试分析阶段从测试角度转化为不同的测试条件，而在测试设计阶段将测试条件转换为测试用例。在测试执行过程中，测试用例执行通过，则通过产品风险与测试用例之间的可追溯性关系，测试团队可以确认与之相关的产品风险是否得到了缓解。测试过程中保持良好的可追溯性，测试经理就可以明确告诉利益干系人，什么需求已经覆盖，哪些产品风险得到了缓解，是否达到了特定的测试目标等。

1.6.3　测试结果比较

测试执行通常是依据测试规程规格说明中定义的测试执行顺序开展的。测试用例的执行既可以通过手动方式进行，也可以是自动化方式进行。对手动测试来说，测试执行主要是测试人员根据测试用例中的步骤进行测试执行，包括测试人员输入必要的测试数据、检查测试输出、比较测试实际结果和期望结果、记录在测试过程中发现的问题等。而对于自动化测试过程，测试执行可能是利用测试工具运行测试脚本，通过自动化的方式记录测试结果。

测试结果的比较是测试用例执行过程中很重要的环节。必须仔细检查每个测试用例执行的实际输出结果，根据测试期望结果判断被测软件产品是否能够正确地工作。测试结果包括中间测试结果和最终测试结果，因此，在测试过程中不仅要对最终结果进行比较，也需要对中间结果进行比较。

假如测试实际输出结果和测试期望结果一致，那么认为该测试用例执行是通过的；假如不一致，需要进一步检查以确定引起不一致的原因。测试结果和期望结果的不一致并不一定是由于测试对象的缺陷引起的，也可能是其他原因造成的，例如，测试用例设计存在问题、测试用例的期望结果存在问题、测试环境存在问题或者测试版本出现错误等。假如是由于测试对象引起的不一致，那么测试人员需要提交相应的缺陷报告。所以，在比较测试结果的时候，需要测试人员从不同的方面来确认具体的问题来源。

测试结果的比较手段可以是手动比较，也可以是通过工具自动比较。手动比较由测试执行人员将测试用例的执行结果和测试用例的期望结果进行比较。手动比较的优点是灵活（不需要固定的描述格式）和自由（可随时选择所需要的测试用例加以执行）；其缺点是效

率低、容易失误。自动比较由系统（例如，自动化测试工具）自动完成测试实际结果和测试期望结果比较的任务。自动比较要求系统对测试用例（或测试数据）有明确的描述方法和比较机制。测试自动比较的优点是效率高和准确性高；它的缺点是灵活度和自由度受测试工具的影响。

1.6.4 确认测试和回归测试

确认测试或再测试是验证缺陷是否得到正确的修复而进行的测试，测试运行的是发现此缺陷的同一个测试用例，测试用例也可能会进行适当的调整。确认测试或再测试的主要目的是确认缺陷的修正是否是有效的。

回归测试是指测试以前测试过并经过修改的软件对象，确保软件变更没有给软件其他未变更部分带来新的缺陷。软件修改后或使用环境变更后要执行回归测试。回归测试在整个软件测试过程中占有很重要的地位，是保证软件质量的一个重要测试活动。回归测试可以应用在各个测试级别：组件测试、集成测试、系统测试和验收测试。

软件开发生命周期中的任何一个阶段，都会发生软件的变更。软件变更之后都需要开展相应的回归测试。可能的变更包括：

（1）缺陷的修复；

（2）版本变更和升级（例如，增加了新的功能或采用了新的技术）；

（3）数据库的变更和升级；

（4）软件使用平台的变更和升级（例如，软件运行环境的变更等）。

在进行回归测试的时候，必须采用合适的回归测试策略确定回归测试的范围。这就涉及回归测试用例选择的策略。下面是几种常用的回归测试策略。

（1）零回归测试：针对缺陷的修复，只做确认测试，即重新运行所有发现缺陷的测试用例，判断新的软件版本是否已经修正了这些缺陷。针对新增功能，只运行所有新增加的功能测试用例，用来判断是否正确实现了新的功能，这是正常测试的一部分。这种策略并没有进行任何回归测试，所以也称为零回归测试。

（2）基于风险的回归测试：是基于风险的分析而展开的，这种方法需要进行变更影响分析。确定变更如何影响现有系统的过程，也称为影响分析，它有助于决定回归测试的广度和深度。回归测试的范围取决于变更影响分析的结果。

（3）完全回归测试：这个策略不考虑变更影响，重新运行所有的测试用例，这是一种安全的回归测试策略，遗漏缺陷的风险最小，但是测试成本很高。

零回归测试只进行了很少的测试，而完全回归测试运行了所有的测试用例，它们在实际测试过程中运用的都比较少，因为零回归测试存在的风险比较高，而完全回归测试工作量巨大。一般来说，基于风险的回归测试在测试过程中运用比较多，由于对软件变更进行了相关的影响分析，测试重点会放在软件变更可能会影响到的功能和模块。在平衡进度、成本和质量的前提下，尽量覆盖风险高的功能和模块，例如，系统中增加了新的功能，需要分析新增加的功能对已有系统的影响，那些和新增加功能存在功能交互的其他功能模块是选择回归测试用例的重点。

1.6.5　测试日志

测试日志是测试过程监控、测试结果和软件产品质量评估的基础，同时也是数据分析和过程改进的重要依据。测试日志的重要作用有：

（1）记录测试过程中所发生的事件；

（2）描述被测系统或组件的测试结果；

（3）作为测试报告的依据。

测试日志的内容随测试级别和策略的不同而改变，例如，如果采用了自动化组件测试，自动化测试工具能自动收集日志信息。如果采用了手工方式测试，测试人员需要手工编写和整理日志。规范或审计的要求有时候会影响日志的内容，如同影响测试实施一样。



1.7　评估出口准则和报告

测试进度监控过程中收集的测试度量信息，可以用来支持和评估测试出口准则，例如，测试用例执行的完成率可以用来判断测试执行的进度。测试评估可以从测试过程和测试质量两个方面进行。通过对测试的评估，可以量化测试过程，判断测试进行的状态，以及判断测试结束的时间。同时，通过测试评估，可以生成一些量化数据，例如，测试覆盖率、缺陷清除率、测试完成率等，它们可以作为软件产品质量评估的输入。

测试评估可以是软件测试过程的一个阶段性的输出，通过生成的测试评估报告来确定测试是否达到出口准则。测试评估应该贯穿于整个软件测试过程，可以在每个测试阶段结束前进行，也可以在测试过程的某个时间点上进行。

评估测试出口准则和报告阶段的主要测试活动有：

（1）将测试状态和测试计划中的出口准则进行比较；

（2）评估是否需要更多的测试执行，或者是否需要更改测试出口准则；

（3）输出测试总结报告。

评估测试出口准则和报告阶段的主要输入有：测试状态报告、缺陷状态报告、风险状态报告、测试周报告/月报告、测试出口准则和测试计划等。

1.7.1　评估出口准则

测试出口准则的评估是检验测试对象是否符合预先定义的一组测试目标和出口准则的活动。测试出口准则的评估可能产生下列结果：测试结果满足所有的出口准则，测试活动可以正常结束；测试出口准则没有满足，要求执行一些附加测试用例；或者测试出口准则要求过高，需要对测试出口准则进行修改。当进行测试出口准则评估时，测试人员必须决定当前测试状态是否完全满足测试计划中的测试出口准则，例如，测试对象有 80%的语句被执行到，可以作为测试出口准则的条目之一。

如果执行完所有的测试用例后，测试出口准则的一个或多个条目还没有满足，那么应

当执行更多的测试用例或者修改测试出口准则。如果需要增加测试用例，需要注意保证新的测试用例满足相应的出口准则。否则，额外的测试用例只会增加工作量，而不会对测试结果有任何改进。

为了满足出口准则要求，有时需要采用不同的测试技术，例如，为了测试被测对象对特定异常输入的响应，而当前的测试环境无法模拟该特定的异常输入，从而导致相应的软件代码无法被执行到。此时，可以结合早期的静态测试（例如代码评审或静态分析），对该代码进行分析和评估。

有时也会出现由于被测对象本身的原因，导致无法满足测试计划中定义的出口准则要求。例如，测试对象中包含死代码，导致无法满足 100%的语句覆盖率。在评估测试出口准则的过程中，要对诸如此类的可能性进行考虑，避免因为无意义的测试出口准则而进行无效的测试。假如测试过程中出现出口准则无法满足时，应该调查无法满足的原因，例如，包含死代码，以期发现更多的缺陷（缺陷屏蔽导致的缺陷）。

除了测试覆盖率，测试出口准则还可以包括其他条目，例如，失效发现率。如果失效发现率下降到给定的阈值（例如，失效发现率小于 0.2 个/人・天），就表明不再需要更多的测试，测试工作可以结束。根据失效发现率评估测试出口准则时还应该考虑失效的严重程度，对失效进行分类，并区别对待。

测试出口准则有时还需要考虑时间和成本方面的因素，例如，被测对象必须在 2016 年 9 月 30 日交付给客户，那么随着交付时间来临，测试活动也只能是强制停止，即使有些测试任务没有完成。此时，测试经理也需要对无法及时完成测试任务进行原因分析，例如，是否是由于项目计划中没有配置足够的测试资源，还是低估了某项测试活动的工作量。

即使在测试过程中消耗比测试计划更多的资源，但是由于更多地发现了测试对象中的缺陷，测试也在一定程度上降低了软件产品的质量风险。通常来说，测试对象中的缺陷导致测试对象在实际运行过程中出现失效而引起的成本，要远远高于在测试时发现并修复缺陷的成本。

完成相关的测试活动之后，测试团队需要向软件产品利益干系人提交测试报告。测试报告中需要明确是否满足测试出口准则，或者罗列尚未满足出口准则的具体条目。不同的测试级别，其输出的测试报告形式可能是不一样的，例如，对于组件测试这样低级别的测试，测试报告的形式可能仅仅是向项目经理汇报关于是否达到出口准则的一个简单的信息；而在相对高级别的测试中（例如系统测试），可能需要输出正式的测试报告。

1.7.2 测试报告

测试报告指的是对软件产品或组件在测试过程中产生的行为及结果进行描述的文件。测试报告以文档的形式描述了被测对象的测试情况和测试结果，并对测试结果和数据进行分析，向项目管理层提供信息和建议。测试报告是测试过程的一个重要输出，必须得到项目管理层的批准才能够成为正式的测试文档。

不同的测试级别，其测试报告的内容可以是不同的，其报告的名称也可以是不一样的，例如，组件测试报告、集成测试报告、系统测试报告、验收测试报告。这些报告在提交人、读者、报告产生的阶段、报告的关注点、报告的依据和报告审核人方面各不相同。

测试报告中描述的结论来自各个测试活动的记录文档，而不是凭空得出的。测试报告需要参考的文档主要有：测试计划、测试设计规格说明、测试用例规格说明、测试规程规格说明、缺陷报告和测试日志等。测试报告需要对度量数据进行分析，例如，测试用例执行度量数据、缺陷度量数据等。

IEEE Std 829—2008 测试文档标准中的级别测试报告和主测试报告的内容构成可以作为编写测试报告的参考信息，也可以根据组织的具体文档模板和项目特性选择测试报告的组成内容。任何一个测试级别完成以及测试工作全部完成以后都需要输出测试报告。

报告的详细程度和频繁程度（次数）取决于项目和组织。这应该在测试计划阶段进行协商，包括与相关项目利益干系人沟通。

1.8 测试结束活动

所有的测试执行活动完成并输出测试报告后，并不意味着测试活动全部结束。当确定测试结束后，应当收集主要的输出成果，并且交给相应的人员或归档，这些活动称为测试结束活动。测试经理和测试团队中的其他成员需要将测试工作产品归档，同时对测试过程和测试活动进行相关数据的收集和分析，总结测试过程和测试活动的经验教训，例如，测试活动是否实现了测试计划设定的目标、有哪些非期望的事情和风险发生、发生的原因是什么、是不是有效地解决了这些风险、是否存在没有解决的变更请求等。

测试结束活动中相关数据的分析主要用来回答下面这些问题：测试过程中哪些方面做得不够好、为什么会存在这些问题、哪些方面做得比较好及相应的经验、如何在下一个项目中能够做得更好等。这些内容的分析可以让测试团队成员了解测试过程中的经验和教训，从而帮助测试团队在以后的测试中尽量避免重复以前的错误。同时，这些经验教训也可以帮助其他项目和其他项目团队改进他们的开发和测试过程，以及提高产品的质量。通过对测试过程的评估（包括测试任务、所花费资源和所达到结果的鉴定评估），可以发现哪些方面需要进一步改进。把这些发现结果使用在以后的项目中，可以帮助后继项目的持续改进。

测试结束活动主要包括以下 4 个方面：

（1）确保所有的测试工作全部完成。例如，所有计划的测试都已经执行；提交的缺陷已经修改，并且进行了相应的确认测试和回归测试；遗留缺陷都经过项目团队的风险分析，认为在当前版本不进行该缺陷的修复而存在的风险是可以接受的，或者当前的资源限制无法解决这个缺陷，确定这些缺陷需要留到下一个版本解决。

（2）移交测试工作产品。例如，延期的或者无法解决的缺陷需要和使用软件产品的用户进行沟通；将测试文档和测试环境等移交给后续进行维护测试的小组。

（3）总结经验教训。记录开发过程和测试过程中所有的经验教训，并且将经验教训文档化，以避免在以后的项目和项目测试中重复这些错误，例如：

① 由于在测试的后期发现不曾预料的缺陷集群。测试团队分析之后发现，假如在早期的风险识别会议上邀请更加广泛的业务干系人来参加，就可以减轻或者避免这类风险的发生。

② 实际的测试工作量和原来估算的工作量差距很大。分析工作量估算误差大的原因，例如，遗漏某些测试工作，在以后的工作量估算中，将这些因素考虑在内，不断提高测试估算的精度。

③ 缺陷趋势的分析、引起缺陷的原因和影响的分析。例如，是不是由于项目后期的变更请求影响了开发的质量；是否由于采用了不好的实践，例如，裁减了某个测试级别，而这个测试级别可以较早地发现缺陷，从而提高测试效率、降低修改成本和节省测试时间；是否由于使用了新的技术、测试人员发生变动以及缺少相应项目的技能。

（4）在配置管理系统中归档所有的结果、记录、报表和其他文档及交付物。例如，测试计划和测试规格说明应该进行归档。

上述测试结束活动非常重要，而在实际测试过程中却常常被遗漏。因此，应该将测试结束活动明确包含在测试计划中。遗漏上述测试结束活动中的一项或多项是常见的事，其原因是多方面的，例如，测试人员过早地被分配到下个软件产品测试；下一个软件产品测试的资源或进度的压力、测试团队过于疲劳等。对于按照合同开展的软件产品，例如，客户定制开发的软件产品，合同中应该明确上述的测试结束活动必须得到有效的开展。

小结

软件测试作为一个系统化的过程，它不仅是软件开发生命周期中的一个阶段，而是应该贯穿于整个开发过程。本章描述了测试过程的组成：测试计划、测试监控、测试分析、测试设计、测试实施、测试执行、评估出口准则和报告，以及测试结束活动。

测试计划应该在软件项目启动时就开始，并贯穿于整个测试生命周期。测试计划阶段主要描述了测试策略的选择、确定测试优先级、定义测试方法、确定测试文档之间的关系、确定开发工作产品和测试工作产品之间的关系、确定测试范围、确定测试环境、了解外部依赖关系和定义测试出口准则等测试活动。

为了高效地对测试过程进行控制，测试经理必须建立项目层面的监督框架。测试过程中收集和分析各种度量信息，可以帮助测试经理判断当前的测试是否满足测试计划的要求。假如当前测试状态和进度信息，与测试计划之间的要求出现偏差，测试经理必须采取相应的手段对测试活动进行控制。而测试监控得到的反馈信息也有助于更新测试计划。

测试分析阶段的主要任务是确定“测试什么”，即识别测试条件。本章主要描述了影响测试条件详细化的因素、测试条件详细化的优点、缺点和适用场景，以及测试条件简单化的适用场景。测试分析过程中有效识别测试条件，邀请技术干系人和业务干系人的积极参与是非常重要的。

测试设计阶段的主要任务是确定“如何测试”，即将测试分析阶段识别的测试条件，通过测试计划中定义的测试技术转化为测试用例。测试设计过程中，测试人员创建、维护和更新可追溯性有助于评估测试覆盖率、测试进度和变更的影响分析，例如，测试依据、需求、测试条件、产品风险和测试用例之间的可追溯性。同时，测试经理还需要明确测试设计是按照概要测试用例还是详细测试用例的形式作为输出。

测试实施阶段的主要任务是设置测试执行优先级、准备测试数据、测试用例自动化、

检查测试执行入口准则，以及确定测试执行进度。同时阐述了尽早进行测试实施的优点和面临的挑战。

测试试行阶段的主要任务是测试人员按照测试规程规格说明中定义的测试优先级运行测试用例，比较测试实际结果和期望结果，假如两者之间出现不一致，分析是由于被测对象引起的问题还是属于误报缺陷。假如是被测对象的问题，测试人员需要提交缺陷报告。在开发修复缺陷之后，测试人员需进行确认测试和回归测试，并将测试执行过程中的各种输出和结果，以测试日志的形式进行记录。

评估出口准则和报告阶段的主要任务是将测试状态和出口准则中的要求进行比较，以确定是否可以结束测试，还是需要增加更多的测试，并最终输出测试报告。

测试结束活动阶段主要描述了 4 个方面的活动：确保所有的测试工作已经完成、将测试工作产品移交到维护团队、开展经验教训总结活动，以及将测试过程中的测试工作产品进行归档。测试结束活动很重要，但测试实践过程中往往由于各种原因而被忽略。

模拟题

1. CTAL-ATM_LO-1.2.1

TM-1.2.1(K4)为了计划测试活动和工作产品以实现测试目标，必须对一个系统的测试需求进行分析。

问题：

你是旅游信息手机应用项目的测试经理。近期该项目切换到敏捷流程和测试驱动开发。每个开发周期持续时间为 15 天，在第 7 天之后开始每日构建。第 10 天以后，不会再有新的功能加入。开发团队由经验丰富的团队成员组成，他们以自己的工作为荣，但对测试团队不太友好。以粗略的用户故事形式编写需求，如下所示。

> US 03-30：查找距离最近的匹配的酒店
>
> 作为不熟悉当地环境的临时客户，希望获得与经济和舒适度要求最匹配且距离最近的酒店信息。
>
> 优先级：高；工作量：7（10 当中的 7）

该软件依赖于已有的 Web 服务，在开发阶段通过桩代替。开发人员负责组件测试，而系统测试和验收测试是测试团队的职责。开发周期早期进行的系统测试，经常由于新开发功能存在严重问题而被阻塞。分析显示，很多此类问题应该在组件测试时就被发现。通过分析运行环境中发现的缺陷显示：30%的性能问题是由于第三方交付的 Web 服务不可靠造成的。

主要的测试目标：缓解性能风险，以及“优先级≥高”的用户故事不会出现高严重程度的失效，从而提高信心。另外，高层管理团队要求测试与开发之间紧密合作。

下面哪三个测试活动/工作产品可以最好地帮助达到测试目标？

选项：

A．在系统测试期间，测试人员将“优先级≥高”的用户故事的性能测试自动化，并在第 10 天开始测试执行

B．在第 10 天之前，参与由开发人员和测试人员开发的组件测试用例的非正式评审

C．项目管理层和测试管理层与服务提供商一起识别外部的 Web 服务，并签署服务级别协议（SLA）

D．在第 10 天之前，开发人员在组件测试过程中完成“优先级=极高”的用户故事的性能测试

E. 测试经理为组件测试定义度量组，并在第 7 天将该度量信息报告给测试经理
F. 在每个开发周期之前测试经理定义集成测试级别计划，并在第 10 天提交给开发人员
G. 在第 7 天之前，当每日构建开始时，测试团队通过审查活动，批准详细设计规格说明

解释：

A. 正确。必须安排性能测试；在第 10 天之前系统还不稳定。
B. 正确。可减少组件测试后阻塞测试的失效数，同时改进开发人员和测试人员的沟通。
C. 正确。报告出来的 30%的性能问题与 Web 服务相关。这（或者部分）可能是由于没有定义 SLA 造成的；
D. 不正确。可以进行性能测试，但是 30%的问题只能在使用实际服务的系统级别发现。
E. 不正确。组件测试应该是开发团队负责的。
F. 不正确。这里没有集成测试级别。
G. 不正确。TDD 从组件测试用例设计开始；敏捷过程通常不提供详细的设计规格说明；

分值：3 分

2. CTAL-ATM_LO-1.3.1

TM-1.3.1(K3)使用可追溯性检查与测试目标、测试策略和测试计划相关的已定义测试条件的完整性和一致性。

问题：

你是旅游信息手机应用项目的测试经理。近期该项目切换到敏捷流程和测试驱动开发。每个开发周期持续时间为 15 天，在第 7 天之后开始每日构建。第 10 天以后，不会再有新的功能加入。开发团队由经验丰富的团队成员组成，他们以自己的工作为荣，但对测试团队不太宽容。以粗略的用户故事形式编写需求，如下所示。

> US 03-30：查找距离最近的匹配的酒店
> 作为不熟悉当地环境的临时客户，希望获得与经济和舒适度要求最匹配且距离最近的酒店信息。
> 优先级：高；工作量：7（10 当中的 7）

该软件依赖于已有的 Web 服务，在开发阶段通过桩代替。开发人员负责组件测试，而系统和验收测试是测试团队的职责。开发周期早期进行的系统测试，经常由于新开发功能存在严重问题而被阻塞。

分析显示，很多此类问题在组件测试时就应该被发现。通过分析运行环境中发现的缺陷显示：30%的性能问题是由于第三方交付的 Web 服务不可靠而造成的。

主要的测试目标是缓解预想的性能风险，以及“优先级≥高”的用户故事不会出现高严重程度的失效，从而提高信心。另外，高层管理要求测试与开发之间紧密合作。

针对验收测试已经定义了下面的出口准则。

AC1：对于“优先级 = 极高”的用户故事，并发数不超过 1000 时软件的响应时间≤3s。

AC2：对于“优先级≥高”的用户故事，并发数不超过 10 000 时软件的响应时间≤10s。

AC3：在系统测试和验收测试时，“优先级≥高”的用户故事不能存在严重的失效。

AC4：所有的用户故事至少使用一个用户验收测试用例覆盖。

测试策略中，针对“优先级≥高”的用户故事的系统和验收测试，要求采用等价类划分技术。

针对当前的开发周期，选择和实现了下面的用户故事（US）。

（P 指优先级，E 指估算的工作量）

US 02-10：针对选择的酒店，播放视频 (P：中等；E: 4)

US 02-20：播放背景音乐 (P: 低；E: 2)

US 03-20：查找 5 个最近的酒店 (P: 极高；E: 4)

US 03-30：查找最近的匹配酒店 (P: 高；E: 7)

针对系统测试的测试分析刚刚开始，识别了如下的测试条件（TC）。

TC 02-10-1：使用各种支持的格式播放视频。

TC 03-20-1：列出最近的 5 个酒店，针对地点采用等价类划分技术。

TC 03-30-1：列出最近的匹配的酒店，针对用户概况和地点采用等价类划分技术。

TC PE-xx-1：针对用户故事 US 03-30，执行 10 000 并发用户请求的性能测试。

TC PE-xx-2：针对用户故事 US 03-20，执行 1000 并发用户请求的性能测试。

为了满足本周期的出口准则，至少还需要增加多少个测试条件？

选项：

A．2

B．1

C．3

D．4

解释：

A．正确。

（1）缺少针对用户故事 US 03-20，执行 10 000 个并发请求允许的响应时间小于等于 10s 的性能测试。

（2）缺少针对用户故事 US 02-20 的测试条件。

B、C 和 D 选项都不正确。

分值：2 分

3. CTAL-ATM_LO-1.3.2

TM-1.3.2(K2)解释可能影响特定测试条件详细程度的因素，以及细化测试条件的优点和缺点。

问题：

下面哪两个因素更能指出测试条件细化对系统测试的必要性？

选项：

A．测试依据质量低下
B．测试设计与执行被外包
C．测试依据经常发生变更
D．测试设计阶段可以咨询领域专家
E．测试条件用于管理里程碑的汇报

解释：

A．正确。详细描述测试条件可以替代质量低下的测试依据，同时有助于缺陷预防。
B．正确。如果测试条件是比较粗略的，则在回答测试条件的改进问题时要付出高昂的代价。
C．不正确。详细测试条件难以维护，参见大纲。
D．不正确。领域专家可以回答测试设计中粗略测试条件的改进问题。
E．不正确。管理层通常不关心详细级别的内容。

分值：1 分

4. CTAL-ATM_LO-1.4.1

TM-1.4.1(K3)使用可追溯性检查所定义的测试条件和设计的测试用例间的完整性和一致性。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

你项目公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

你项目的公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

考虑在质量风险分析过程中识别出下列风险项：**为公司网站设计的定制化注册按钮，无法链接到公司积分与奖励忠诚度计划的 URL**。

假设你已经通过可追溯性确定逻辑测试用例已经覆盖该风险项。

下面哪个正向逻辑测试用例是完整的、正确的，并且覆盖了该风险项？

选项：

A．单击公司注册按钮，验证你可以链接到公司的积分与奖励计划的注册页面

B．单击公司注册按钮，验证你可以链接到我们公司的首页

C．快速单击公司注册按钮，查看会发生什么

D．单击通往首页的 URL，检查是否显示首页

解释：

A．正确。根据场景，其输入将会得到正确的期望结果，并且与该风险项相关。

B．不正确。错误的期望结果。

C．不正确。可能覆盖了该风险项，但它是一个负面测试用例，并且没有包含期望结果。尽管它可以作为该风险项进行探索性负面测试的一个较好选项。

D．不正确。这是一个很好的正面逻辑测试用例，但是它不能覆盖该风险项。

分值：2 分

5. CTAL-ATM_LO-1.5.1

TM-1.5.1(K3)使用风险、优先级、测试环境和数据依赖以及限制条件，制定测试执行的进度，该进度与测试目标、测试策略和测试计划保持完整和一致。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

你项目公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

你项目的公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

考虑在质量风险分析过程中识别出下列风险项：**为公司网站设计的定制化注册按钮，无法链接到公司积分与奖励忠诚度计划的 URL**。

假设技术方面的项目干系人评估该风险的可能性为中等。

仅依赖于上面给定的信息，下面哪个论述肯定是正确的？

选项：

A. 该风险的严重程度，应该评估为最高级别
B. 与该风险相关的测试用例，应该在测试执行阶段首先执行
C. 与该风险相关的测试用例，应该在测试执行阶段的中期进行
D. 基于该风险项的严重程度，应该有大量的测试用例与之关联

解释：

A. 正确。该风险涉及该应用的核心功能。
B. 不正确。严重程度最高和可能性最高的测试用例应该在此测试之前进行。
C. 不正确。该论述不确定是否肯定正确，因为不知道该风险项与其他风险项之间的关系。
D. 不正确。该论述不确定是否肯定正确，因为不知道基于严重程度与可能性的组合，如何确定工作量的分配。

分值：2 分

6. CTAL-ATM_LO-1.6.1

TM-1.6.1(K3)使用可追溯性监督测试进展与测试目标、测试策略和测试计划的一致性和完整性。

问题：

某公司的测试采用基于风险的测试策略。该项目目前处于测试执行阶段。针对下面的测试用例，分别提供了测试用例 ID、风险级别、测试用例覆盖的需求 ID 和当前的测试状态。

测试用例 ID	风险级别	需求 ID	状态
02.007	非常高	09.003	失败
02.010	高	09.003	准备执行
02.019	非常低	09.020	通过

下面哪两个论述是正确的？

选项：

A. 测试团队可能没有遵循测试策略，因为测试用例 02.010 的风险级别高于 02.019
B. 假如测试计划要求尽可能早的为每个需求至少执行一个测试用例，则执行的顺序可能是正确的
C. 测试执行顺序肯定不正确，因为测试用例 02.010 的风险级别高于 02.019

D. 测试经理应该停止测试执行，针对当前测试顺序中存在的所有问题进行评估
E. 执行测试用例 02.019 是浪费时间，因为它并没有发现任何缺陷

解释：

A. 正确。基于风险的测试策略中，高风险的测试应该在低风险的测试之前执行。
B. 正确。与 02.010 相比，02.019 覆盖了不同的需求。
C. 不正确。可能如选项 B 中的情景，或者也可能是由于测试用例被阻塞，因此不按照风险顺序执行。
D. 不正确。虽然按照测试顺序评估问题是有意义的，但是没有必要在评估的同时停止测试执行。
E. 不正确。发现缺陷不是测试的唯一目的。
分值：2 分

7. CTAL-ATM_LO-1.7.1

TM-1.7.1(K2)解释在测试过程中准确和及时收集信息的重要性，以便支持准确的报告和对照出口准则进行评价。

问题：

针对组件测试执行，下面哪两个度量最应该包括在测试进度报告中？

选项：

A. 计划的与实际达到的覆盖率
B. 计划的与实际报告的缺陷
C. 组件测试的缺陷发现百分比(DDP)
D. 识别的测试条件数目
E. 组件测试时间与集成测试时间

解释：

A. 正确。根据“ISTQB 软件测试人员认证高级大纲-测试经理”模块内容。
B. 正确。根据“ISTQB 软件测试人员认证高级大纲-测试经理”模块内容。
C. 不正确。无法及时测量。
D. 不正确。与测试分析进度相关。
E. 不正确。无法及时测量。
分值：1 分

8. CTAL-ATM_LO-1.8.1

TM-1.8.1(K2)概述 4 组测试结束活动。

问题：

下面哪一项最准确地描述了“测试完成检查”这项测试结束活动？

选项：

A. 测试完成检查应确保所有计划中规定的测试工作已经完成
B. 测试完成检查应确保所有重要的经验教训已文档化
C. 测试完成检查应确保所有的测试工作产品已存储在配置管理系统中
D. 测试完成检查应确保建立计划以保证所有良好实践能够重复

解释：

A. 正确。参见大纲章节 1.3。
B. 不正确。
C. 不正确。
D. 不正确。
分值：1 分

9. CTAL-ATM_LO-1.8.2

TM-1.8.2(K3)进行项目回顾以评价过程和发现改进领域。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

你项目公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

你项目的公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

假设该项目已经完成初始版本发布，公司和客户已经使用该系统一个月。你的团队采用基于风险、基于需求和应对型的混合测试策略。在质量风险分析过程中，对按钮定制化的评估结果认为它是最低风险区域，而注册属于最高风险区域。你正在进行测试工作的经验教训回顾会议。

下面哪三个区域应该是该回顾会议的关注点？

选项：

A. 是否已经识别和修复了影响公司或客户使用的注册功能相关的缺陷

B. 针对注册功能、按钮定制化功能和积分管理功能，比较实际完成的测试用例与估算的测试用例之间是否存在差异

C. 评估是否覆盖了按钮定制化功能相关的质量风险，并提交了发现的严重问题

D. 将已知缺陷和失败的测试用例移交给系统支持团队

E. 确定项目计划中是否包括所有相关的项目风险，它们会影响该产品的及时交付，并影响最早使用该产品的公司

F. 确定针对注册功能、按钮定制化功能和积分管理功能测试用例所需的详细程度

G. 测量注册功能相关的需求覆盖率，并将之汇报给项目和业务利益干系人

解释：

A. 正确。注册功能是核心需求区域，测试回顾会议应该检查在基于需求的测试策略下，是否在该区域有缺陷遗漏。

B. 正确。在回顾会议中，应该检查产品的主要功能区域是否低估了所需的工作量。

C. 正确。在回顾会议中，分析缺陷信息以评估质量风险分析是否正确。

D. 不正确。这属于测试结束活动的一部分，但不属于回顾会议的内容。

E. 不正确。这是回顾会议中很好的一个考虑点，但它是项目层面的问题，不是测试相关的问题；

F. 不正确。这应该属于测试实现阶段。

G. 不正确。这属于测试控制的范畴。

分值：2 分

第2章 测试管理

本章学习目标如表 2-1 所示。

表 2-1　学习目标

编　　号	学习目标描述	级　　别
TM-2.2.1	分析软件项目或程序的干系人、环境、需求，包括软件开发生命周期模型，并识别最佳测试活动	K4
TM-2.2.2	理解软件开发生命周期中的活动和工作产品如何影响测试，以及测试如何影响软件开发生命周期中的活动和工作产品	K2
TM-2.2.3	解释在基于经验的测试和非功能性测试中对测试管理问题进行处置的方法	K2
TM-2.3.1	基于风险的测试及其他测试优先级设定和工作量分配的方法	K2
TM-2.3.2	举例说明产品风险评估的不同技术	K2
TM-2.3.3	分析、识别和评估产品质量风险，从关键项目干系人的角度总结风险及评估的风险级	K4
TM-2.3.4	描述在生命周期和测试过程中，怎样根据评估的风险级别，适当地缓解和管理识别的产品质量风险	K2
TM-2.3.5	举例说明测试选择、优先级设定和分工的不同方案	K2
TM-2.4.1	分析给出的测试方针和测试策略，建立主测试计划、级别测试计划和其他与这些文档相补充和相一致的测试工作产品	K4
TM-2.4.2	针对给定的项目，分析项目风险并选择适当的风险管理方案（如缓解、应急、转移和/或接受）	K4
TM-2.4.3	描述并举例说明测试策略如何影响测试活动	K2
TM-2.4.4	制定适合组织、生命周期以及项目需要的测试工作产品的文档规范和模板，适时裁剪标准的主体部分，生成可用模板	K3
TM-2.5.1	针对给定的项目，使用所有适当的估算技术创建整个测试过程活动的估算	K3
TM-2.5.2	理解可能影响测试估算的因素，并举例说明	K2
TM-2.6.1	描述并比较典型的测试相关度量	K2
TM-2.6.2	比较不同层面的测试进度监控	K2
TM-2.6.3	分析和汇报测试结果，主要包括遗留风险、缺陷状态、测试执行状态、测试覆盖状态以及信心来提供见解和建议，帮助项目干系人做出发布决策	K4
TM-2.7.1	分别给出决定质量成本 4 种类别的例子	K2
TM-2.7.2	基于质量成本，以及其他定性和定量的考虑，估算测试的价值，并将其告知利益干系人	K3
TM-2.8.1	理解如何成功运用不同人员策略构建团队：是利用分布式团队、外包团队还是内包团队	K2
TM-2.9.1	总结软件测试标准的来源及使用	K2

本章相关术语如表 2-2 所示。

表 2-2　术语

英　　文	中　　文	说　　明
level test plan	级别测试计划	通常用于一个测试级别的测试计划。参见 test plan
master test plan	主测试计划	通常指针对多个测试级别的测试计划。参见 test plan
product risk	产品风险	与测试对象有直接关系的风险。参见 risk
project risk	项目风险	与（测试）项目的管理与控制相关的风险。例如，缺乏配备人员、严格的限期、需求的变更等。参见 risk
quality risk	质量风险	有关质量特性的产品风险。参见 quality attribute, product risk
risk	风险	可能会导致负面结果的因素。通常表达成可能的（负面）影响。[GBT 11457]
risk analysis	风险评估	通过评估识别出的风险，以估计其影响和发生的可能性的过程
risk assessment	风险评估	参见 risk analysis
risk identification	风险识别	使用技术手段（例如，头脑风暴、检查表和失效历史记录）标识风险的过程
risk level	风险级别	风险的重要性，由风险的影响和可能性定义。风险级别用于决定测试的强度。风险级别既能用定性的词（比如，高、中、低）表示，又能用定量的词表示
risk management	风险管理	对风险进行标识、分析、划分优先级和控制所应用的系统化过程和实践[GBT 11457]
risk mitigation	风险缓解	参见 risk control
risk-based testing	基于风险的测试	在项目初始阶段使用的一种测试方法，用来降低产品风险的级别，并通知干系人产品风险的状态。该方法包括产品风险识别和使用风险级别指导测试过程
test approach	测试方法	针对特定项目的测试策略的实现，通常包括根据测试项目的目标和风险进行评估之后所做的决策、测试过程的起点、采用的测试设计技术、退出准则和所执行的测试类型
test conditions	测试条件	组件/系统中能被一个或多个测试用例验证的条目或事件。例如，功能、事务、特性、质量特性或者结构化元素
test control	测试控制	当监测到与预期情况背离时,制定和应用一组修正动作以使测试项目保持正常进行的测试管理工作。参见 test management
test director	测试总监	管理测试经理的高级经理，参见 test manager
test estimation	测试估算	对测试中消耗的工作量、完成时间、涉及成本、测试用例数等近似计算的结果
test leader	测试组长	参见 test manager
test level	测试级别	统一组织和管理的一组测试活动。测试级别与项目的职责相关联。例如，测试级别包括组件测试、集成测试、系统测试和验收测试（与 TMap 一致）
test management	测试管理	计划、估算、监控和控制测试活动，通常由测试经理来执行
test monitoring	测试监督	处理与定时检查测试项目状态等活动相关的测试管理工作。准备测试报告来比较实际结果和期望结果。参见 test management
test plan	测试计划	描述预期测试活动的范围、方法、资源和进度的文档。它标识了测试项、需测试的特性、测试任务、任务负责人、测试人员的独立程度、测试环境、测试设计技术、测试的进入和退出准则和选择的合理性、需要紧急预案的风险，是测试策划过程的一份记录（与 IEEE 829 一致）[GBT 9386]

续表

英　　文	中　　文	说　　明
test policy	测试方针	描述有关组织测试的原则、方法和主要目标的高级文档
test strategy	测试策略	一个高级文档，该文档定义了需要对程序（一个或多个项目）执行的测试级别和需要进行的测试
Wide Band Delphi	宽带德尔菲法	一种专家测试评估的方法，旨在集中团队成员的智慧进行精确的评估

2.1 简介

随着测试工作经验的积累，测试人员将会规划不同的测试职业发展方向，例如，测试经理。本章主要关注测试人员在晋升为测试组长、测试经理及测试总监时所需的具体测试管理知识和技能。尽管不同的组织对这些职位和担任这些职位的人员的职责等级有不同界定，但本书将这些专业测试人员统称为测试经理。本章的主要内容如下。

（1）一定条件下的测试管理：主要包括与测试相关的主要项目利益干系人、其他软件开发生命周期活动及工作产品、测试活动与软件开发生命周期其他活动是如何集成的、如何有效管理非功能性测试、如何管理基于经验的测试等内容。

（2）基于风险的测试和其他测试优先级设定和工作量分配的方法：主要包括基于风险的测试、基于风险的测试技术、测试用例选择的其他测试技术、测试优先级设定和工作量分配的其他技术。

（3）测试文档和其他工作产品：描述了与 IEEE Std 829—2008 兼容的测试方针、测试策略、主测试计划和级别测试计划，以及探讨了如何在测试过程中有效管理项目风险，其他与测试活动相关的工作产品等内容。

（4）测试估算：描述了测试估算的基本概念以及影响测试估算的各个因素。通过具体的项目案例详细地分析各种可行的测试估算技术，例如，基于团队的测试估算、基于百分比的测试估算、基于测试规模的测试估算等。

（5）定义和使用测试度量：分别从风险、测试（或测试用例）、覆盖率、缺陷和信心 5 个方面，阐述测试进度的监控活动，并讲解如何利用在监控活动中观察到的测试过程相关的信息和问题，制定应对计划控制当前的测试过程，并提供改进建议。

（6）测试商业价值：从质量成本入手，讲解了预防成本、检测成本、内部失效成本和外部失效成本的基本概念。并从定量分析和定性分析两个方面分析具体的质量成本。

（7）分布式测试、外包测试和内包测试：简单介绍了分布式测试、外包测试和内包测试的基本概念，并对这三种测试类型的优缺点进行了分析。

（8）管理行业标准的使用：主要阐述了与测试活动紧密相关的国际标准、国内标准和特定行业标准，以及各类型标准主要关注的侧重点。

测试管理除了上面罗列的内容之外，还涉及评审、缺陷管理、测试过程改进、测试工具与自动化、团队建设等，这些内容将在后续的章节独立展开描述。

本章以及后续的章节将会以一个实际的项目案例为基础，对测试管理中涉及的各种测试文档、技术和方法、管理技能、工具和自动化等进行详细阐述。以下是该项目的简单描述。

iBAS 系统是通信领域中的智能宽带接入服务器。iBAS 是城域网接入层和边缘汇聚层之间的智能业务网关，对以太网、ADSL 接入用户进行身份认证、授权、计费，对其业务施加控制策略，保证用户业务和运营商网络的安全性，同时提供其他增值业务。

iBAS 系统是一个系列化的以太网接入设备，在用户侧通过 LANSwitch、ADSL 接入等二层设备接入以太网、WLAN 等宽带用户，并对所接入的宽带用户实现用户隔离、用户接入控制、用户访问控制和计费等管理功能，同时可以支持网络地址转换、组播等业务，适用于电信运营商在居民小区、商用大楼、宾馆、机场和学校等场所开展的宽带业务。图 2-1 是 iBAS 系统的组网图。

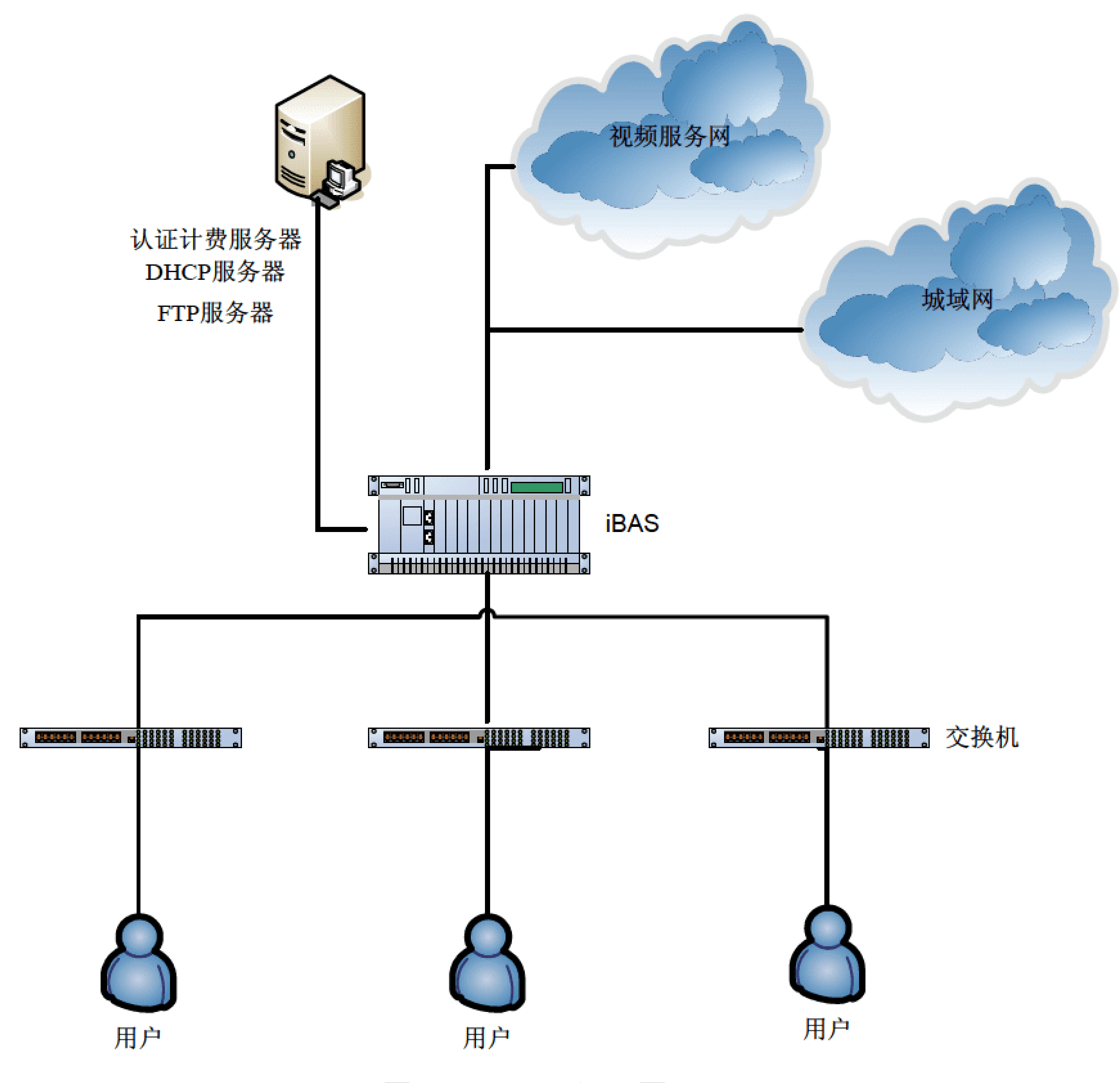


图 2-1　iBAS 组网图

iBAS R1.0 是 iBAS 产品的第一个版本，主要提供如下功能。

（1）提供以太网接入。上行接口：支持 100M/1000M 以太网；下行接口：支持 100Base-T、100Base-FX。

（2）支持 RADIUS（Remote Authentication Dial In User Service，远程用户拨号认证服务）认证、计费，支持基于流量、连接时间、分时段等多种计费方式。

（3）支持静态地址和通过 DHCP（Dynamic Host Configuration Protocol，动态主机配置协议）动态获得 IP 地址。

（4）支持 PPPoE（Point to Point Protocol over Ethernet，基于以太网的点对点协议）用户接入。

（5）通过 VLAN（Virtual Local Area Network，虚拟局域网）划分，实现用户二层隔离，

限制 VLAN 下接入用户数量。

（6）为接入用户提供组播业务功能 IGMP（Internet Group Management Protocol，因特组管理协议）。

（7）支持基于用户级别的流量控制。

（8）提供标准的 SNMP（Simple Network Management Protocol，简单网络管理协议）接口，支持集中网管系统。

iBAS R1.0 项目包括的功能较多，本书中主要涉及其中的 IGMP 功能。关于这个功能的系统需求描述，请参见附录 A。

2.2 一定条件下的测试管理

测试管理活动贯穿于整个软件测试生命周期，即第 1 章中描述的基础测试过程。测试经理必须通过合理的测试计划和测试过程监控，保障软件项目的成功，或者防止出现严重的失效，从而体现测试价值。

测试管理必定是在特定条件下开展的，因此测试经理必须根据其他项目利益干系人的需求、环境和资源、活动（如软件开发生命周期的阶段和任务）、工作产品（如需求规格说明）等要求和特点，适当地安排测试过程，包括测试活动及工作产品。本章节主要内容如下。

（1）了解利益干系人；

（2）软件开发生命周期其他活动及工作产品；

（3）测试活动和软件开发生命周期其他活动的整合；

（4）管理非功能性测试；

（5）管理基于经验的测试。

2.2.1 了解利益干系人

测试活动贯穿于整个软件开发生命周期，因此测试人员需要与软件项目其他人员交流沟通、协调互动和反馈监控。本章将他们统称为利益干系人，有效识别利益干系人有助于更高效地开展测试活动。

利益干系人是指与测试活动、测试工作产品、最终软件系统或交付物的质量有利益关系的人。利益干系人的利益可以表现为直接或间接地介入测试活动、直接或间接地接受测试工作产品，或者直接或间接地受到软件项目输出的交付物的质量的影响。利益干系人的类型受到软件项目、产品、组织或其他因素的影响，主要包括：

1．开发人员、开发组长和开发经理

该利益干系人的主要职责是根据软件产品设计要求实现和部署软件代码；然后根据测试团队测试结果，例如发现的缺陷，有针对性地定位、重现和修复缺陷。开发人员对缺陷进行修复并验证完成之后，再将代码集成后发布给测试团队进行确认测试，以及回归测试。

2．数据库架构师、系统架构师和设计师

该利益干系人的主要职责是设计软件架构、数据库架构或者系统架构，并作为开发人

员编码的重要参考输入。假如测试团队在测试过程中发现的问题是由于设计原因导致的，那他们需要分析其中原因并进行修复。

3．市场和业务分析师

该利益干系人的主要职责是确定软件产品必须呈现的特性，以及这些特性的固有的质量水平要求。他们通常还参与确定测试范围，评审测试结果，根据测试结果对产品质量做出评估，并基于这些信息做出决策。

4．高层管理人员、产品经理和项目发起人

该利益干系人的主要职责是参与确定测试范围，评审测试结果，并根据测试结果做出决策。

5．项目经理

该利益干系人的主要职责是负责管理软件项目，并最终确保项目取得成功。为了达到这个目标，项目经理在项目管理过程中，需要平衡产品特性、质量、进度和成本。他们通常还需要负责获取测试活动所需的资源，并协同测试经理进行测试计划和监控。

6．技术支持、客户支持和帮助台人员

该利益干系人的主要职责是针对用户和客户在使用软件产品过程中碰到的问题和疑问，为他们提供技术支持，同时他们也是已交付软件产品的功能与质量的受益人。

7．直接和间接用户

该利益干系人直接使用软件产品，即最终用户，或者接受由该软件产品生产或支持的输出或服务。

根据软件产品、开发模型和组织等的不同特点，测试经理必须识别符合当前软件项目或者软件产品的具体利益干系人。测试经理必须确切理解不同利益干系人与测试之间的本质关系，以及测试团队如何更好地满足利益干系人的需求。

除了需要识别上述的利益干系人之外，测试经理还应该识别影响软件测试的其他活动和工作产品，或者受到测试影响的软件开发生命周期其他活动和工作产品。只有更好地理顺测试活动和工作产品，与软件开发生命周期其他活动和工作产品之间的关系，才能更好地提高测试效率和测试的有效性。

2.2.2　软件开发生命周期其他活动及工作产品

软件测试通常是在软件开发生命周期的各种活动的大背景下开展的，而软件测试的一个重要目标是评价由该生命周期生成的软件工作产品的质量。因此，测试经理必须了解其他活动和工作产品怎样影响测试，并在了解测试怎样影响其他活动和工作产品的情况下，计划和指导测试活动。

以敏捷开发实践的组织为例，开发人员执行测试驱动开发，生成自动化的单元测试用例，然后不断将代码，以及针对这些代码的测试集成到配置管理系统中。在开展测试驱动开发过程中，测试经理应与开发经理共同确保将测试人员融合到这些活动中，并与这些活动保持一致。测试人员可以通过评审单元测试用例，为提高测试覆盖率和测试活动的有效性提出自己的建议，同时深入了解该软件功能以及实现情况。测试人员也需要想办法将他们自己创建的自动化测试，尤其是功能回归测试用例集成到配置管理系统。

组织特点、软件产品特点、软件开发模型等因素都会影响测试活动、利益干系人、软件开发生命周期活动和工作产品之间的关系。但下面的软件开发生命周期活动与测试活动之间的关系尤其密不可分。

1. 需求工程和管理

需求规格说明是测试生命周期中各个测试活动的主要参考依据（即测试依据），因此需求工程和管理将影响每个测试活动的有效开展。例如，测试计划阶段，需求规格说明可以帮助确定测试范围和估算测试工作量；测试分析与设计阶段，需求规格说明中的需求是识别测试条件和获取测试判断准则的重要依据；测试执行阶段，通过运行测试用例，判断软件是否正确实现和满足了需求规格说明中的功能和要求，同时可以基于需求开展需求覆盖率的评估。

同时，缺陷修复或者功能增强贯穿于软件开发生命周期中，特别是采用增量-迭代开发模型时，而这都会导致需求发生变更。而需求变更必定会影响测试工作产品做出相应的变更，因此测试经理需要对需求变更保持高度的警惕。测试过程中保持测试用例和需求之间的可追溯性，有助于测试经理更好地应对需求变更。当需求出现变更时，基于可追溯性可以帮助测试经理采取合理的测试控制活动以应对这些变更，例如，通过需求变更的影响分析，确定回归测试的深度和广度。

测试分析师和技术测试分析师参与需求工程和管理是非常重要的，他们需要参与前期的需求评审，基于需求识别测试条件和设计、执行测试用例，并通过执行测试用例验证需求是否正确得到实现，假如没有正确实现，需要提交相应的缺陷报告。

2. 项目管理

软件项目启动时，测试经理提供测试工作量的数据，以指导测试计划的制定。而测试工作量的估算，需要测试分析师和技术测试分析师的协助。基于测试工作量的估算数据，测试经理将测试进度和测试资源等需求提交给项目经理。测试经理必须与项目经理共同面对项目计划的变更，并采取测试控制活动以适应这些变更。

需要注意的是，项目管理和测试管理是相互影响的，例如，项目计划的变更，会导致测试计划相应进行变更；同样，测试计划的变更，也会影响项目计划。

3. 配置和变更管理

配置管理的目的是在整个软件开发生命周期内，建立和维护软件产品（例如，组件、数据和文档等）的完整性。

测试经理必须与测试团队一起确定测试对象版本交付流程和机制，并记录在测试计划中。测试经理可以要求测试分析师和技术测试分析师生成版本验证测试，并在测试执行过程中确保版本控制。对测试而言，采用配置管理可以确保：

（1）识别被测对象的所有相关项，确保其版本受控，并且相互之间有关联以及和开发项（测试对象）之间有关联的变更可跟踪，从而保证可追溯性；

（2）在测试文档中，可以清晰明确地引用所有标识的文档和软件项。

对于测试人员来说，配置管理可以帮助他们唯一地标识并且复制测试项、测试文档、测试用例和测试用具。

4. 软件开发和维护

测试经理应与开发经理一起协调测试对象的交付，包括每次测试所需的软件版本发布

内容与发布日期，以及参与到缺陷管理中（详细内容请参考第 4 章）

5．技术支持

测试经理应与技术支持经理一起确保在测试结束时测试结果能顺利交付，以便软件产品发布后参与产品支持的人员知道软件产品还存在哪些失效，以及为这些失效所采取的变通方法。另外，测试经理应该与技术支持经理一起，对测试过程中发现的缺陷和用户反馈的信息进行分析，以实施测试过程改进。

6．编写技术文档

测试经理应与技术文档经理一起确保测试文档能按时交付，且这些文档中体现了对缺陷的管理。

2.2.3　测试活动和软件开发生命周期其他活动的整合

通过软件开发模型管理所有的活动，可以使得软件开发工作系统化且处于可控状态。软件开发模型有多种，例如，瀑布模型、V 模型、螺旋模型以及其他各类增量-迭代模型，还有目前流行的“敏捷”或者“轻量级”开发模型。所有这些模型都定义了一种系统化的方式，以达到工程项目中工作的有序化。下面对这些常见的开发模型做一个简单的介绍。

1．顺序模型

顺序模型主要包括：瀑布模型、V 模型、W 模型等。图 2-2 是一个典型的 V 模型框架图。

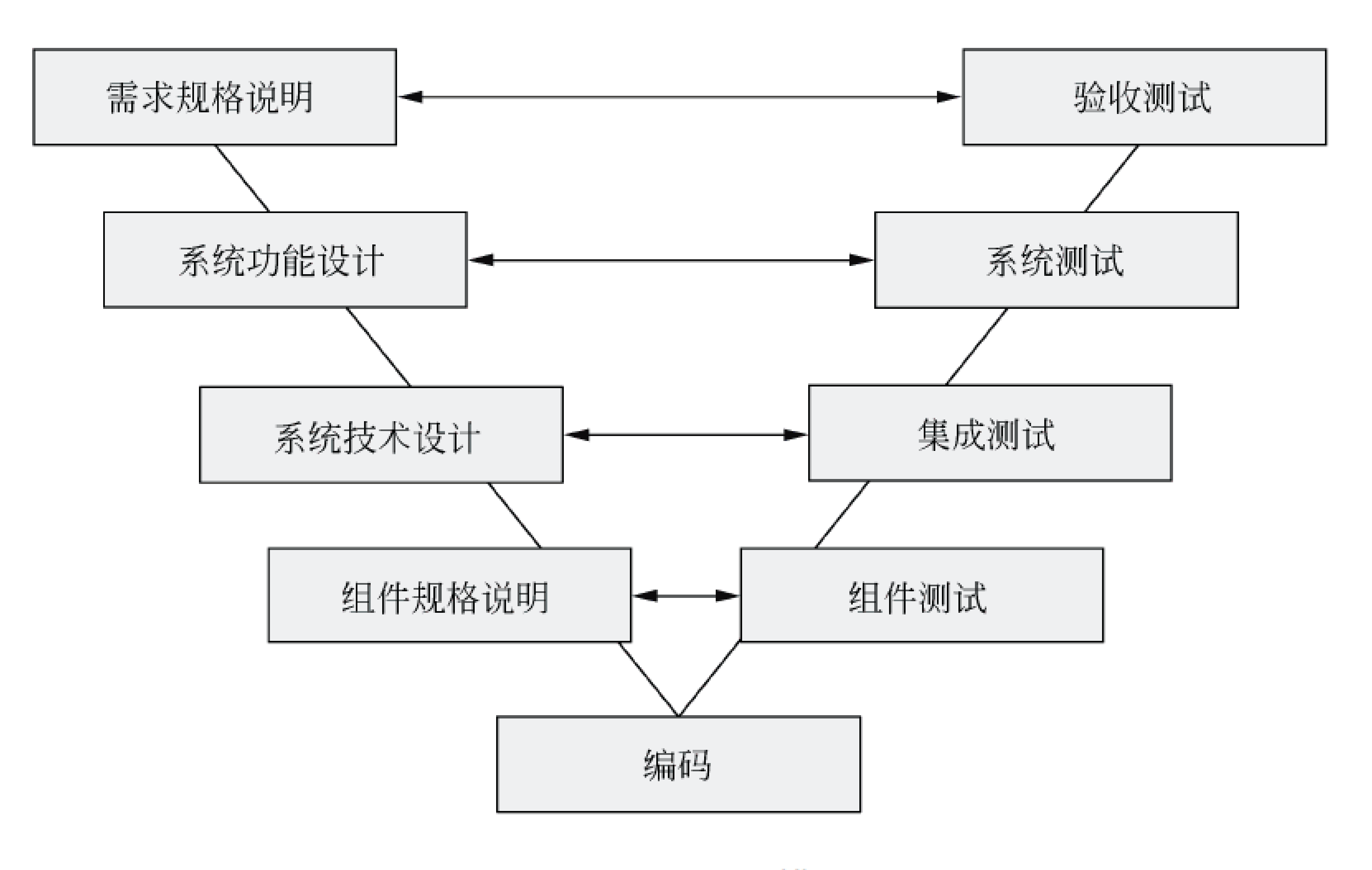


图 2-2　通用 V 模型

理论上，顺序模型各阶段的活动和输出的工作产品应该是串行的（如需求、设计、实施、单元测试、集成测试、系统测试和验收测试），即前面阶段的输出是后面阶段的输入，完成前面的阶段之后才开始下个阶段的任务。测试计划、测试分析、测试设计和测试实施与项目计划、业务/需求分析、软件和数据库设计、编程并行进行，具体如何并行则取决于该测试的级别。测试执行按照 ISTQB 基础级大纲和高级测试经理大纲中讨论的测试级别的顺序进行。

顺序模型的特点是每个阶段是顺序进行的，其成功实施的前提是软件产品具备完善明确的需求。但随着软件开发和产品的不断发展，人们很难在项目开始时就进行完善的需求分析和设计，这就导致顺序模型下不断进行返工，而增量-迭代模型有助于改善这些不足。

2. 增量-迭代模型

常见的增量-迭代模型有快速应用开发（Rapid Application Development，RAD）和统一开发过程（Rational Unified Process，RUP）。增量-迭代模型的基本特点是：根据软件产品的业务优先级或者风险，将待实现的特性进行分组；针对每个特性组，分别开展和实施每个项目阶段活动和任务，包括每个阶段的工作产品。

在项目计划和需求分析的同时，测试团队可以进行粗略的测试计划和测试分析等测试工作。详细的测试计划、测试分析、测试设计和测试实施发生在每次增量-迭代之初并行进行。测试执行通常包括不同的测试级别。每个测试级别都应尽早开始，在后续更高级别的测试开始后也可能会继续前面低级别的测试。

3. 敏捷开发模型

敏捷开发模型，如 Scrum 和极限编程（eXtreme Programming XP），这些也属于增量-迭代开发模型，但相对于上面提到的增量-迭代模型，它们的周期更短，通常为 2～4 周时间。每次增量-迭代的活动和工作产品，需要在下次增量-迭代开始之前完成，即每个增量-迭代是顺序进行的。

敏捷开发模型下的测试活动开展方式，与上述的增量-迭代模型类似，但测试活动与开发活动之间存在更大程度的重叠，包括各测试级别的测试执行与开发活动之间的重叠。每个增量-迭代的所有活动，包括测试活动，都应在下个增量-迭代开始之前完成。在敏捷开发模型中，测试经理的角色通常从直接的管理人员转变成了技术权威/咨询顾问。

4. 螺旋模型

螺旋模型通过软件项目早期使用原型来确认项目的可行性、试验设计和实施决策。螺旋模型根据业务优先级和技术风险，选择原型试验开展的顺序。螺旋模型也是增量-迭代开发模型的一种，它兼顾了快速原型迭代的特征以及瀑布模型的系统化与严格监控。螺旋模型最大的特点在于引入了其他模型不具备的风险评估，使软件在无法排除重大风险时有机会停止，以减小损失。在每个增量-迭代阶段构建原型是螺旋模型用以减小风险的途径。通过对原型进行测试，以确定该软件相关的技术问题的哪些方面仍然没有得到解决。假如该技术问题得到解决，就可以按照顺序模型或增量-迭代模型继续实施项目。

无论采用何种软件开发模型，软件测试都应是整个软件开发生命周期的重要组成部分。测试经理了解各种开发模型有助于将测试活动集成到不同的软件开发生命周期模型中。下面以 V 模型的系统测试级别为例，阐述 ISTQB 基础测试过程与软件开发生命周期是如何有效集成的。

（1）系统测试计划与项目计划同时开始，测试监控贯穿于整个软件开发生命周期，即测试监控一直持续到系统测试执行和测试结束工作完成。

（2）系统测试分析和设计与需求规格说明、系统和架构（概要）设计说明、组件（详细）设计说明的编写同时进行，测试团队积极参与相关测试依据的评审是该阶段的重要活动之一。

（3）系统测试实施活动可在系统设计期间开始，但通常情况下都是与编码和组件测试

同时进行。系统测试相关的部署工作通常会延续到系统测试执行开始前的几天。

（4）当满足所有的系统测试入口准则时开始系统测试执行活动。系统测试入口准则一般是指至少完成了组件测试，通常还包括完成组件集成测试。系统测试会被持续执行直到系统测试出口准则被满足为止。

（5）整个系统测试执行期间都在评估系统测试出口准则和汇报系统测试结果，一般来说，当项目即将到达最后期限时评价和汇报的频率更高，也更紧迫。

（6）在满足了系统测试出口准则和正式宣布系统测试执行工作完成之后，进行系统测试结束活动，结束活动有时也可以推迟到验收测试结束或所有项目活动完成后进行。

除了顺序开发模型（例如 V 模型），在增量-迭代生命周期模型中，同样必须将测试活动集成到该过程中，尽管其任务开展的时间、深度和广度等会有所不同。例如，顺序开发模型中会在项目初期部署整个测试环境，而在增量-迭代-迭代开发模型中，更适合部署本次增量-迭代所需的测试环境，以提高测试效率。不管是顺序模型还是增量-迭代开发模型，都需要尽早开展测试计划活动。

除了测试计划阶段，测试执行和报告也会受到开发模型的影响。例如，增量-迭代开发模型中，下个增量-迭代开始之前，首先完成编写本次周期的完整报告，以及开展增量-迭代结束的评审会议是比较有效的，即建议将每次增量-迭代看成一个小型项目。项目组可以根据上个增量-迭代中识别的缺陷和经验教训，在下个版本中做相应的纠正和调整。

增量-迭代模型的每个周期相对比较短，因此需要缩短在测试评估和报告等活动上的时间，同时将该任务的开展作为跟踪整个测试进度和识别风险的一种方式。如果测试团队没有开展根本原因分析和采取纠正措施，那么某个增量-迭代中发生的问题不仅容易影响下个增量-迭代，甚至在下个周期中会再次犯类似错误。

测试经理应该在测试计划期间，对各测试级别和选定的开发模型与测试过程的组合进行项目特定的集成。其他关于将测试活动和软件开发生命周期其他活动集成的信息，也可以参考测试策略章节（见 2.4.2 节）中的内容。

除了典型 V 模型中的 4 个测试级别之外，根据不同组织、项目和产品的特点和需求，有时还需要增加不同的测试级别（ISTQB 基础级大纲中并没有正式定义），例如：

（1）硬件-软件集成测试；

（2）系统集成测试；

（3）特性交互测试；

（4）客户产品集成测试。

测试过程采用的测试级别类型和数目，会随着组织、产品、项目和开发模型等的不同而不同。但是每个测试级别都应明确定义以下要素。

（1）测试目标。不同测试级别，其测试目标会有所不同，例如，验收测试的主要目的是增加信心，确认软件产品可以按照预期进行工作。

（2）测试范围和测试项。

（3）测试依据，以及衡量该测试依据覆盖率的方式（如可追溯性）。

（4）测试入口准则和出口准则。

（5）测试交付物，包括测试结果汇报。

（6）合适的测试技术，以及测试技术覆盖率的评估方式。

（7）与测试目标、测试入口准则和出口准则以及结果汇报相关的度量和度量标准（包括覆盖率的度量）。

（8）针对具体的测试任务应用的测试工具（适用时）。

（9）测试资源（如测试环境）。

（10）测试团队成员的角色和职责的定义。

（11）符合组织、法规或其他规范的要求（假如适用）。

组织层面需要对上述要素在不同测试级别进行一致的定义，以避免出现遗漏或重复，从而导致风险或者浪费资源。相关内容，可以参考 2.4 节内容。

2.2.4 管理非功能性测试

ISTQB 高级测试经理大纲中关于质量特性的讨论，采用了 ISO/IEC 9126 质量模型中的内容（ISO/IEC 25000 系列标准已经取代 ISO/IEC 9126，但是当前的 ISTQB 高级测试经理大纲还是采用 ISO/IEC 9126 标准的内容。因此，本书将继续以 ISO/IEC 9126 内容作为质量特性的参考）。ISO/IEC 9126 标准中的质量特性又可以分成多种不同的产品质量特性，即质量子特性。

表 2-3 中列出了 ISO/IEC 9126 中包含的质量特性和它们的子特性，以及测试分析师和技术测试分析师所关注的特性和子特性。更多内容，参见高级技术测试分析师大纲第 4 章 [ISTQB ATTA SYL]。

表 2-3 测试分析师和技术测试分析关注的质量特性

特　性	子　特　性	测试分析师	技术测试分析师
功能性	准确性，适合性，互操作性，依从性	X	
	安全性		X
可靠性	成熟度（健壮性），容错性，易恢复性，依从性		X
易用性	易理解性，易学性，易操作性，吸引性，依从性	X	
效率	性能（时间行为），资源利用性，依从性		X
维护性	易分析性，易改变性，稳定性，易测试性，依从性		X
可移植性	适应性，易安装性，共存性，易替换性，依从性		X

ISO/IEC 9126 标准中涉及的质量特性和子特性是开展功能测试和非功能性测试的分析和设计活动的基础，尽管测试经理不需要参与测试分析和设计等技术活动，但是了解相关的质量特性可以帮助测试经理更好地进行测试计划、监督、控制和管理。

假如测试经理没有足够的软件产品业务能力处理非功能性测试计划的任务，可以将部分非功能相关的活动委派给技术测试分析师，或者寻求测试分析师的支持。测试经理在考虑非功能性测试时一般需要考虑下面的一些因素。

（1）利益干系人需求：测试依据中的非功能性需求定义通常都很粗略，有时甚至根本没有提供。测试计划阶段，测试人员必须有能力从受影响的利益干系人那里获得他们对软件产品非功能性要求的期望，并评估这些期望所代表的风险。

（2）所需的测试工具：非功能性测试通常需要特定测试工具的支撑，商业工具或模拟器与性能、效率和安全性测试密切相关。测试计划应包括对使用测试工具的成本和时间的

估算。同时应考虑到新测试工具的学习过程或雇用外部工具专家的成本。

（3）特定的测试环境：许多非功能性测试需要仿真的测试环境，以便进行实际的测量。被测对象的规模大小和复杂度会对测试计划时间和成本预算产生很大影响。执行非功能性测试的成本可能非常高，例如，为了验证访问量很大的互联网网站是否符合可扩展性需求，可能要模拟数十万的虚拟用户，这对硬件和工具的成本有很大影响。

（4）组织因素：非功能性测试可能包括测量软件产品不同组件的行为（例如，服务器、数据库、网络）。若这些组件分布在不同地点和组织中，则需要花很大的工作量进行计划和协调，例如，某些软件组件可能只在每天或一年的特定时间段才能执行系统测试，或组织只能提供有限的时间用于支持测试。假如未能确认其他相关组织能否参与测试，就会严重扰乱预订的测试计划。

（5）安全性要求：在测试计划阶段就应考虑到用于确保系统安全的手段，以保证所有的测试活动可行，例如，使用数据加密技术可能增加创建测试数据和验证结果的难度。数据保护政策和法令可能会禁止基于生产数据生成虚拟用户，那么创建匿名测试数据将是一件重要的任务，必须将其作为测试实施的一部分。

尽管有测试分析师或技术测试分析师的支持，测试经理在管理非功能性测试过程中还是会碰到一些问题。

首先，假如测试经理没有对非功能性测试进行良好的计划，可能会导致软件产品发布之后出现严重的，甚至是灾难性的软件产品质量问题。然而，由于时间和资源的限制，以及非功能性测试的特殊性，例如，很多非功能性测试是非常昂贵的，因此测试经理必须依据风险选择需要执行的非功能性测试，并按照风险评估结果进行测试执行优先级的选择。另外，非功能性测试有多种类型，例如，效率测试、可移植性测试等，测试经理应该根据软件产品特点选择合适的非功能性测试类型。

其次，测试经理需要考虑如何将非功能性测试集成到整个软件开发生命周期中去。而测试经理往往会等到完成所有的功能测试之后再开始非功能性测试，其结果是导致太晚发现关键的非功能缺陷，使得定位和修复缺陷的成本太高。因此，尽早在测试计划中对非功能性测试按照优先级安排，并根据风险优先级进行测试执行排序是很重要的。同时，将非功能性测试的活动贯穿于整个软件开发生命周期。通常在早期的测试级别，甚至是在开发期间，通过多种途径缓解非功能风险。例如，在软件系统设计期间，积极参与用户界面原型的易用性评审，对于识别重大的易用性缺陷是十分有效的。假如到系统测试结束时才发现这些缺陷，可能会导致项目进度的严重延后，修复的成本也将大大增加。

第三，增量-迭代开发模型中开展非功能性测试有其特殊性。由于该开发模型中每个增量-迭代的周期相对更短，甚至比开展非功能性测试的设计和实施活动的时间还短。而其变更和迭代速度会让测试人员难以关注非功能性测试，因为它们通常需要构建复杂的测试框架和测试环境。在这种情况下，一种可行的策略是将非功能性测试活动，脱离每个开发迭代成为独立的测试工作，即非功能性测试活动贯穿于几个连续的迭代时间周期中。

2.2.5　管理基于经验的测试

基于经验的测试是系统化测试的一个有效补充，可以高效地发现其他测试技术遗漏的

一些缺陷，因此它常常用来检查其他测试技术的完整性。基于经验的测试更多地依赖于测试人员的知识、直觉和经验，因此测试人员的经验和能力的不同，对测试结果的完整性和正确性影响很大。

尽管基于经验的测试会带来灵活性和更高效的优点，但也会给测试管理带来一系列的挑战。例如，由于基于经验的测试一般采用轻量级测试过程，且通常早期测试准备很少，因此很难评估该测试技术达到的覆盖率；由于基于经验的测试会弱化测试文档的输出，因此测试经理需要特别留意测试结果的可重复性，尤其是当多个测试人员同时参与测试活动时。

常见的基于经验的测试技术有错误推测法和探索性测试。错误推测法的一个结构化方法是列举可能的缺陷，并设计测试用例攻击这些缺陷，这种系统的方法称之为缺陷攻击。测试人员可以根据经验、已有缺陷和失效数据等方面的技能来设计和维护缺陷和失效列表。本章节内容将主要关注探索性测试。

探索性测试是指依据包含测试目标的测试章程，而测试设计、测试执行、测试记录和学习是同时进行的，并且是在规定时间内完成测试。探索性测试是一个不断交互的过程，根据当前测试执行的结果，调整后续的测试设计和测试执行活动。在探索性测试的开始，并没有像传统的测试方法那样对所有的测试活动和测试任务进行计划和设计，而是根据测试活动的不断开展，动态调整后面的测试活动和测试任务。这里的测试设计和测试执行并没有严格的顺序关系，各个活动之间互相影响，动态调整。

探索性测试是一个特殊的测试过程，它的测试活动和测试内容是动态变化的，更多的是通过测试执行的结果来指导后续的测试活动，花在文档上的时间较少，这也就意味着探索性测试的可管理性不强，对于每个测试人员执行的测试活动的进度和效果很难监控。为了更好地开展探索性测试，Jonathan Bach 提出了“基于会话的测试管理”（Session-Based Test Management，SBTM）方法[①]，该方法可以对探索性测试进行更好的管理，把探索性测试的优势更好地发挥出来。

基于会话的测试管理中的会话（Session）指的是测试工作的一个基本单位，在每个会话中测试工作是连续进行的，不会被打断。会话中的测试活动是为了实现某个具体的目标，同时测试活动的结果是可评审的。任何可能会被电话、会议、电子邮件或其他非测试活动打断的测试活动，都不适合成为一个会话。在一个会话中，测试人员应该专注于测试活动而不被打扰。在每个会话结束的时候，应创建描述已进行的测试活动的报告，该报告通常称为会话单（Session Sheet）。

测试经理需要对该会话的报告或者会话单进行评审。一个会话可以持续 90 分钟，也可以是三个小时，对于会话的持续时间并没有严格的要求，通常情况下，每个测试人员一天完成的测试会话不要超过三个。由于每个会话持续的时间比较短，所以对于每个会话进行报告的周期也相应比较短，但是内容也相对简洁。Jonathan Bach 将一个测试会话中的任务分成以下三类。

（1）创建会话：建立测试环境和增进对软件产品的理解。

（2）测试设计和执行：仔细检查测试对象，寻找问题。

① Jonathan Bach. Session-Based Test Management. Software Testing and Quality Engineering magazine, 11/00.

（3）缺陷调查和报告：测试人员发现被测对象疑似问题时，需要检查是否是由于被测对象原因导致的失效，假如是被测对象导致的问题，需要提交缺陷报告。

上述的三个任务并不一定是顺序执行的，在实际的操作中经常都是重叠的。每个测试会话完成后，测试人员要填写会话单。基于会话的测试管理中的会话单主要内容如下。

（1）会话章程（Session Chapter）：主要描述会话的任务和可能采用的策略，会话的章程可能是测试人员自己选择的，也可能是测试经理分配的。

（2）测试人员姓名。

（3）会话起始的日期和时间。

（4）任务分解（以会话的形式）：通常分别描述为创建会话、测试设计和执行、缺陷调查和报告。

（5）数据文件：描述测试会话中使用的测试数据文件。

（6）测试备注：对于测试活动中的任何特别的说明都可以记录在这里。

（7）问题：描述测试过程中碰到的问题，例如，由于测试环境不足，有些测试没有执行。

（8）缺陷：描述测试过程中发现的缺陷。

测试经理根据测试人员提交的会话单，在测试团队内进行任务报告会议。在任务报告会议中，测试经理回顾测试活动、改善测试章程、从测试人员处取得反馈、估算和计划进一步的会话。任务报告的议程主要是简要地检验以下各项内容，简称 PROOF，并明确下一步的测试会话章程。

（1）经历（Past）：在会话期间发生了什么？

（2）结果（Result）：会话取得了什么成果？

（3）展望（Outlook）：还有什么有待解决的问题？

（4）障碍（Obstacles）：要进行良好的测试有些什么障碍？

（5）感受（Feeling）：测试人员对上述内容有什么感觉？

探索性测试的测试对象和内容比较发散，经常出现具体的测试活动和会话章程不一致的情况。那么在最后的会话单中要对会话章程进行修改，以保证提交的会话单如实地反映了实际的测试活动。如果需要的话，也可以新建一个新的会话来记录这次测试活动。虽然不建议会话的持续时间太长，但是在实际的测试过程中，仍然会出现一个会话占用测试人员一整天的情况，例如，测试环境的搭建或者缺陷的调查花费的时间比预期的要长。这种情况下也没有必要强行把会话拆成多个更小的会话，可以根据测试的需要保留这些持续时间较长的会话。

测试会话最好是在没有被打断的情况下持续地进行，但有时候可能需要参加一个紧急的会议，或者开发人员修复紧急的缺陷而需要测试人员的帮助，这个时候测试会话不得不被打断。在这种情况下，测试会话的中止和恢复是允许的。

除了基于会话的测试管理之外，探索性测试管理的另一种方式是将自主和自发的探索性测试集成到更为传统的预先设计好的测试过程中，即将探索性测试和脚本化测试进行合理的平衡。测试人员在获取预先定义的测试步骤、测试输入和期望结果的基础之上，开展更多的探索性测试活动。测试人员也可以在开展脚本化测试之前、期间或者之后，开展更加自由的自主测试会话，并将它们作为每日测试的组成部分。假如在探索性测试会话中发

现了缺陷，或者有值得后续进行更深入测试的内容，测试人员需要对预定义的测试进行更新。

2.3 基于风险的测试和其他测试优先级设定以及工作量分配的方法

软件测试的一条基本原则是，穷尽测试是不可能的，因此测试经理必定会面临如何在有限时间和资源下开展测试活动的问题。另外，测试过程中也经常会碰到这样的情况：开发团队发布测试版本出现了延迟，导致测试活动常常处于时间和资源的高压状态。此时，将测试活动和测试任务等进行优先级划分，将测试资源分配到最重要的部分是测试经理常常可以采用的一个有效测试策略。

推动测试工作高效有序地开展，是测试经理经常在测试过程中面临的一个挑战。测试经理必须以合适的策略实施测试用例的选择、分配和优先级设定，即测试团队在几乎无限的测试条件和组合中选择一组有限的测试条件，确定适当的分工以确保测试用例覆盖各个测试条件，按照一定的策略对生成的测试用例设定测试优先级，然后以优先级顺序执行测试用例，并对测试结果和产品质量进行评估。

后续章节将通过识别和分析风险以及其他因素，阐述如何平衡有限时间和资源与几乎无穷测试之间的矛盾，以更好地实施该测试策略。由于测试过程中各种资源的限制和其他因素的相互制约，各种策略之间需要相互折中和平衡。

2.3.1 基于风险的测试

风险是指负面或不希望发生的后果或事件发生的可能性。假如可能发生如下问题：测试利益干系人对软件产品的质量，或者软件项目是否能够及时发布，产生怀疑甚至没有信心时，就存在风险。

不同的风险，其风险级别是不一样的。风险级别由两个因素决定：问题发生的可能性（风险的可能性）和问题发生后的影响程度（风险的严重程度）。一种简单的方法是两者相乘就可以得到风险级别：

风险级别=风险可能性×风险严重程度

风险评估和风险缓解等风险管理活动将基于风险的可能性和严重程度展开。根据风险的潜在影响的不同，可以将风险分为两类：项目风险和产品风险。

项目风险，有时称为计划风险或者管理风险，指的是潜在问题主要影响的是软件产品是否能够及时完成，或者是否能够及时发布，例如，具备特定测试技能的测试人员由于各种原因无法及时到位。关于项目风险的管理，可以参考 2.4.5 节的内容。

产品风险，有时称为质量风险或者产品质量风险，指的是潜在问题主要影响的是软件产品的质量。质量包括影响客户、用户和利益干系人满意度的全部功能、行为、特征和属性。产品风险的例子包括：软件产品中的计算错误，即与准确性相关的功能风险；用户输入时间响应慢，即与效率和响应时间相关的非功能风险；用户界面和字段难以理解，即与

易用性和易理解性相关的非功能风险。通过测试发现缺陷，并在发布前修复缺陷，测试就可以缓解产品风险。当通过测试不再能发现缺陷时，即在特定的测试配置和环境下，软件产品可以正常运作，那么通过测试同样也缓解了产品风险，增强对产品质量的信心。

基于风险的测试，其主要的关注点在产品风险上，其过程是邀请利益干系人参与软件产品质量分析，尽量多地识别产品风险，并评估其可能性和严重程度得到风险级别并确定优先级，然后测试团队设计相应的测试用例覆盖产品风险，并实施和执行测试用例缓解产品风险。

尽量多地邀请利益干系人参与风险识别和评估，是成功实施基于风险的测试的关键点。风险识别和评估小组应包括软件产品所有利益干系人的代表，根据需要有些干系人会代理其他干系人，例如，针对大众市场开发的软件产品，可能会抽样部分潜在客户，要求他们识别对他们使用有严重影响的软件缺陷。在这种情况下，抽样的部分潜在客户作为所有最终客户的代理干系人。由于测试人员在产品风险和失效方面的专长，他们应该积极参与风险识别和评估过程。

基于风险的测试采用产品风险选择测试条件，并为测试条件分配测试时间，生成测试用例的优先级。基于风险的测试的主要目标就是通过测试降低软件产品的质量风险水平，即将风险级别降低到可接受的范围。

基于风险的测试包含以下 4 个主要活动。

（1）风险识别；

（2）风险评估；

（3）风险缓解；

（4）风险管理。

在某种程度上，上述风险活动在时间上是连续的，但针对持续的风险管理而言，它们在大多数软件项目中，是指迭代地进行风险管理活动。

1．风险识别

风险识别主要是识别软件开发生命周期中存在哪些风险，并以文档的形式进行记录。根据组织、软件产品和团队技能等的不同，可以采用不同的风险识别方法。不论是产品风险还是项目风险，测试人员都可以通过下列的一项或多项技术识别风险。

（1）专家咨询；

（2）独立评估；

（3）风险模板；

（4）经验教训（例如项目评估会议）；

（5）风险研讨会（例如失效模式和影响分析）；

（6）头脑风暴法；

（7）风险分类（或检查表）；

（8）过去的经验；

（9）问卷调查法。

广泛邀请可能的项目利益干系人（例如系统人员、客户、用户等），有利于在风险识别过程中发现尽可能多地识别重要风险。对参与风险识别的利益干系人的抽样越广泛，风险识别过程就越有可能识别出越多的产品风险。同时，要求参与风险识别的利益干系人，

应该覆盖不同技能的人员参与进来，例如，利用不同人员的技术能力、开发能力、市场能力等共同进行风险的识别和分析。

头脑风暴法是一种常用的风险识别方法。测试经理负责安排和召集相关人员进行头脑风暴，识别测试中可能存在的风险。其目的是得到一份相对全面的测试风险列表，然后对这些风险进行定性分析或者定量分析。参与头脑风暴的成员从各自的经验、角色和职责等方面考虑测试过程中可能存在的风险，以及产生这些风险可能的原因，并将得到的风险按照不同的类型进行分类，使得测试风险更加清晰明了。

头脑风暴法常常可以和其他风险识别方法结合起来使用。下面将通过案例的方式讲述问卷调查法、风险模板、风险分类等方法在风险识别方面的应用。

1）问卷调查法

在风险识别过程中，通过设计问卷调查对相关的风险进行识别是一种常用的方法。

☆示例：风险问卷调查法

表 2-4 是针对 IGMP 功能的风险识别过程中采用的问卷调查表（由于篇幅的原因，只截取了其中的部分，以说明问卷调查法在风险识别中的作用）。

表 2-4　风险问卷调查表

项目名称：IGMP 功能测试　　　　时间：

参与人员：XXX

问卷调查表：

序号	问 卷 问 题	是/否	备　　注
1	需求规格说明是否在开始设计之后还在修改		
2	需求规格说明中是否存在“未确定”的内容		
3	需求规格说明中是否包含你认为应该包含的内容		
4	功能的复杂性是否超出组织目前的经验和技能		
5	需求或者功能是否难以设计		
6	需求的实现是否都有相应的解决方案		
7	功能的内部接口是否清晰地定义（例如：软件与硬件之间）		
8	是否存在性能瓶颈方面的问题（例如：用户加入的时间）		
9	功能设计和实现是否做过性能分析		
10	采用的硬件是否会限制需求的实现（例如：内存容量）		
11	功能是否需要和商业软件 COTS 进行集成		
12	功能是否会进行足够的组件测试（例如：有足够的时间）		
13	假如没有足够时间进行组件测试，组织是否会采用折中的方法		
14	设计规格说明是否足够详细地进行代码和测试脚本的开发		
15	是否有足够的硬件资源进行集成测试和系统测试		
16	是否能够开发客户实际的场景和数据来验证需求		
17	需求中定义的性能要求是否可测试		
18	硬件系统是否能够及时到位		
19	相关的测试仪表和设备是否能够及时到位		
20	测试人员是否能够熟练使用测试设备		
21	需求规格说明中是否有难于验证的需求		

续表

项目名称：IGMP 功能测试　　　　时间：
参与人员：XXX
问卷调查表：

序号	问 卷 问 题	是/否	备　注
22	功能开发和测试是否在多个地方进行		
23	开发团队成员是否有相关的开发经验		
24	开发团队是否熟悉开发过程和测试过程		
25	开发过程的变更控制是否有效		
26	测试计划和相关文档是否随着需求变更而变更		
27	团队成员在不同的功能之间是否能够精诚合作		
28	团队成员之间是否可以顺畅沟通		
29	团队成员是否可以及时得到一些可能影响工作的信息		
30	功能相关人员是否缺乏要求的特殊技能		

测试经理和测试团队通过收集和分析问卷调查表的结果，可以得出一份和测试相关的风险列表，通过对风险进行相应的分析和评估，然后根据结果采取合适的应对措施。

2）风险模板

假如组织或者项目层面已经定义了风险的模板，那么开展相关的风险识别活动就会相对简单。因为在风险模板中，已经针对可能识别的风险进行了合理的分类。而风险的分类是组织或者项目的经验总结，可以为将来项目的风险识别提供比较完善的思路，帮助项目团队更好地开展风险识别活动。

☆示例：风险模板

表 2-5 是应用风险模板，对 IGMP 进行风险识别的结果。

表 2-5　基于风险模板的 IGMP 风险列表

序　号	风 险 描 述	注　释
1	**资源**	
1.1	IGMP 测试仪表无法及时到位	
1.2	IGMP 测试硬件平台缺乏	
1.3	参与 IGMP 测试的人员目前还在其他项目中，无法及时介入 IGMP 测试工作	
1.4	IGMP 测试人员需要参与其他测试任务，例如：产品的入网测试	
1.5	缺乏 IGMP 测试服务器和 PC	
2	**技术**	
2.1	IGMP 测试范围的变更，由于需求文档中缺少非功能性需求的描述	
2.2	测试团队缺乏 IGMP 测试经验	
2.3	IGMP 测试工作量的偏低估算，有些测试活动没有估算在内	
2.4	用户的 IGMP 需求经常发生变更	
3	**质量**	
3.1	系统需求规格说明和设计规格说明评审过程中发现大量的缺陷	
3.2	测试计划没有经过详细的评审	
3.3	测试设计规格说明没有经过详细的评审	

续表

序　号	风 险 描 述	注　释
4	**沟通**	
4.1	需求人员、开发人员和测试人员之间缺乏正规的沟通平台	
4.2	测试人员在测试过程中发现的缺陷，开发人员无法及时到现场进行确认	
5	**第三方/外包**	
5.1	IGMP 相关的驱动部分通过外包方式进行开发，可能无法及时提交，或者提交的质量存在较大的不确定性	
6	**法律**	
6.1	某测试工具软件，测试团队只有两个授权（Licenses）。在测试过程中要保证只有两台机器安装该工具软件，避免不必要的版权纠纷	

在使用组织层面的风险模板时，由于模板中已经提供了风险分类，例如，资源、技术等，因此，可以提供较好的思路来开展风险识别活动。假如风险模板和头脑风暴法结合起来使用，效果会更好。有了风险列表之后，就可以开展后续的风险评估和相关的风险缓解活动。

3）风险分类

风险分类或者风险检查表是在测试过程中识别风险的一个非常有用的工具，可以帮助识别各种可能的测试相关的风险（包括以前项目的测试风险、经验等），例如，资源、技术、沟通等风险。通过风险分类，首先确定潜在的风险列表，然后将这些风险映射到测试对象的不同模块。其中的风险分类可以是组织层面的经验积累，也可以是根据测试人员的经验，通过头脑风暴法得到的。下面是几种不同的风险分类。

（1）质量特性分类

质量特性的设计主要是用来验证需求规格说明中定义的需求。下面以 ISO/IEC 9126 质量模型为基础，得到的质量特性列表可以作为风险识别的一种方法。

① 容量（Capacity）：测试对象是否满足需求定义的容量要求。

② 可靠性（Reliability）：在所有要求的条件下，测试对象是否能正常工作，不出现失效。

③ 易用性（Usability）：客户是否轻松简单地使用产品。

④ 性能（Performance）：测试对象运行的速度和响应时间。

⑤ 易安装性（Installability）：产品是否容易安装到目标运行平台上。

⑥ 兼容性（Compatibility）：测试对象是否可以和外部组件、系统等协同工作。

⑦ 易测试性（Testability）：产品是否可以被有效地测试。

⑧ 可维护性（Maintainability）：产品是否可以经济地构建、修改或者功能增强。

⑨ 易本地化（Localizability）：产品是否可以经济地以另外一种语言进行出版发布。

这个风险列表并不是标准的，具体风险列表中包含的质量特性应该根据测试对象、产品和组织项目特征进行仔细的选择。基于质量特性得到的风险列表可以作为评审需求规格说明的一个重要输入。

（2）常见风险列表

一般来说，常见风险列表对大部分的软件系统或者产品都是适用的。下面是其中的一些典型的风险。

① 复杂性：规模庞大、复杂以及难以理解的对象。

② 新的：产品功能对项目成员都是全新的，没有任何经验可以借鉴。

③ 变更的：变更或者进行了改进的功能对象。

④ 关键的：软件系统或者产品的某个部分的失效会导致严重的后果。

⑤ 精确的：软件系统或者产品的某个部分需要达到严格的精度。

⑥ 常用的：客户经常操作或者使用的部分。

⑦ 特殊的：对于组织特别重要的部分，例如，竞争对手不具备而运营商关注的功能。

⑧ 第三方软件：软件系统或者产品中使用的不是该项目团队开发的部分。

⑨ 分布式的：需要集成工作，但是由不同地点或者团队所开发的部分。

⑩ 缺陷集群区域：已知的缺陷很多的地方。

（3）领域内的风险类别

领域内的风险类别是针对某个特定领域而罗列的风险列表。针对某个风险类别，可以将在测试中碰见的各种问题都放在一起，形成一个风险类别。表 2-6 是针对软件安装测试的风险列表。

表 2-6　软件安装相关的风险

风 险 类 别	详细的风险
安装了错误的文件	临时文件没有被清除
	不需要的文件被安装了
	需要的文件没有被安装
	正确的文件被安装在错误的目录
被破坏的安装文件	旧文件替代了新的文件
	在升级过程中用户数据库被破坏了
其他应用程序被破坏了	和其他产品共享的文件被修改了
	属于其他产品的文件被删除了
硬件没有被正确地配置	
安装操作人员错误地替换或者修改了关键文件或者参数	
安装过程时间过长	
安装过程需要有固定的人员监控	
安装过程使人混淆	用户接口容易误用
	安装信息或者指导步骤不清晰

根据前面描述的不同风险分类，分别得到了基于质量特性的风险列表、常见的风险列表和领域内的风险列表，接下来可以将这些风险列表应用于实际的测试过程中。下面是在实际测试过程中运用风险列表的建议步骤，可以作为将来测试的参考。

① 确定需要分析的对象，是整个产品，还是单独的组件或者几个组件。

② 确定功能或者组件关注的程度。例如，以“高”“中”和“低”分别作为不同的关注程度。产品或者组件中的任何对象，在默认情况下都作为“中”的风险，除非有特别的理由相信它是“高”的风险，或者“低”的风险。利用风险的关注程度，对于分析人员而言是有意义的，但需要注意其中的一些主观成分。

③ 收集分析对象的信息。在进行分析之前，需要了解分析对象的一些情况。例如，按常规风险分类，需要收集哪些相关信息，并在需要的时候，得到相关专家的支持；或者将相关的专家召集到会议室，进行风险评估活动。

④ 检查每一个风险，分析并确定其重要性。针对任何一个风险，都应该确定风险发生的可能性和严重程度，并将这些相关的信息记录在案。

⑤ 假如风险列表中的风险真的发生了，那么将它们记录在案。

⑥ 将任何可能影响风险评估的信息记录在案。

⑦ 检查风险的分布。

风险识别过程中也会识别一些非软件产品质量相关的风险，例如，项目风险。不过项目风险并不是基于风险测试关注的焦点。然而，项目风险管理对于包括基于风险测试在内的所有测试方法都是很重要的，具体内容请参考 2.4.5 节的内容。

2．风险评估

风险识别活动用来尽可能多地识别风险，而风险评估过程中最重要的活动是确定风险的可能性和严重程度，并根据这两个结果得到风险级别。后续的风险缓解和监控活动都是基于风险级别展开的。

风险发生的可能性一般指测试对象中存在潜在问题的可能性。影响风险发生可能性大小的因素包括：

（1）技术复杂性；

（2）业务分析人员、设计人员和程序员的技能水平；

（3）团队内部矛盾；

（4）与供应商的合同问题；

（5）开发团队的地理分布；

（6）老方法和新方法之间的对立；

（7）工具和技术；

（8）不良的管理领导和技术领导；

（9）时间、资源和管理压力；

（10）缺乏初期的质量保证；

（11）高变更率；

（12）初期的高缺陷率；

（13）接口和集成问题。

风险发生的严重程度是指当风险发生后，对用户、客户或其他干系人影响的严重性。影响风险发生的严重程度的因素包括：

（1）使用受影响的特性的频率；

（2）对组织形象的损害；

（3）商业损失；

（4）潜在的金融、生态或社会方面的损失或法律责任；

（5）民事或刑事法律制裁；

（6）失去经营执照；

（7）缺少合理的变通；

（8）明显的失效导致负面宣传。

对识别的风险进行合理分类，是风险评估的另一个重要组成部分，例如，ISO/IEC 9126 标准中列出的质量特性就可以作为风险分类的参考。有的组织有自己的质量特性的定义。风险分类有时候也可以作为风险识别活动的一个组成部分，例如，在使用检查表作为风险识别的基础时，风险类型的分类会在风险识别过程中进行。

测试人员可以通过定量或定性的方法确定风险发生的可能性和严重程度。如果风险的可能性和严重程度可定量确定，则可将两个值相乘，计算得到风险级别（即可能性 P 和严重程度 S，风险级别 $RL = P \times S$）。但是对可能性与严重程度进行量化是很困难的，因此有时通过定性的方式会更加合适。即可以将可能性定性为：非常高、高、中等、低和非常低。同样地，可以将严重程度定性为：非常高、高、中等、低和非常低。但其缺点是无法用精确的百分比表达出来。表 2-7 是风险级别定性分析的一个例子。

表 2-7　风险级别的定性分析

风险严重程度	**风险可能性**				
	非常低	**低**	**中等**	**高**	**很高**
非常高	低	中等	高	非常高	非常高
高	低	中等	高	高	非常高
中等	非常低	低	中等	高	高
低	非常低	非常低	低	中等	中等
非常低	非常低	非常低	非常低	低	低

尽管定性分析相对比定量分析容易，但是不能认为定性分析的方法比定量分析的方法差。实际上，如果错误地使用定量分析方法，也会误导利益干系人对于风险级别的理解和管理。为了方便对风险进行分析和应对，可以将定性的风险评估和定量的风险评估相结合，分别将定义的风险可能性的 5 个级别分配不同的数值。例如：

（1）非常低：1。
（2）低：2。
（3）中等：3。
（4）高：4。
（5）非常高：5。

同时也将定义的风险严重程度的 5 个级别分配不同的数值。例如：

（1）非常低：1。
（2）低：2。
（3）中等：3。
（4）高：4。
（5）非常高：5。

将定性的风险评估和定量的风险评估结合之后，尽管得到的风险可能性和严重程度是以定性的方式标识，但在具体的分析和计算过程中，可以将风险发生的可能性和严重程度用对应的数值进行代替，得到一个量化的风险等级，从而方便风险级别的确定，以及和风险阈值进行比较。假如计算得到的风险级别大于定义的风险阈值，那么就需要采取相关的风险缓解措施，以降低风险级别（例如，降低风险发生的可能性或者严重程度，或者两者兼而有之）。

除非风险评估是基于广泛有效的风险数据系统（例如保险业），否则无论是定性的风险评估还是定量的风险评估，它都是基于对风险发生可能性和严重程度的直觉判断。也就是说，个人对于这些因素的理解和看法会直接影响风险级别的确定。不同的利益干系人，例如，项目经理、程序员、用户、商业分析人员、系统人员和测试人员等，对于风险可能

性与严重程度有不同的理解和看法。风险评估过程应包括一些策略和方法，可以使得风险级别在不同利益干系人之间达成一致。否则，风险级别将不能用于指导风险缓解活动。

3．风险缓解

针对风险识别得到风险列表，通过定性分析或者定量分析得到不同风险发生的可能性和严重程度，并计算得到它们的风险级别。根据计算得到的不同风险级别，接下来就需要对识别的风险采取合适的缓解措施。下面是缓解风险经常采用的策略。

1）风险减轻

风险级别由风险发生的可能性和严重程度两个因素决定，因此，通过降低风险可能性或者严重程度（或者同时降低可能性和严重程度）就可以降低风险级别。将风险级别降低到一个可以接受的阈值（风险阈值），那么就成功地实现了风险的应对，例如：

（1）质量低下的系统需求规格说明的风险。测试团队可以在早期采取严格的评审活动，尽早和尽量多地发现和修复其中的缺陷，从而降低该风险级别。

（2）测试团队中缺乏具体的领域知识的风险。可以在项目前期开展软件产品知识和技能培训活动，或者安排测试人员进行需求规格说明和相关标准的学习和研究，以降低这一风险的级别。

2）风险避免

可以将风险避免看作是风险减轻的一种特殊情况，即将风险可能性或者严重程度降低为 0，就可以达到风险避免的目的，例如，iBAS R1.0 项目中 IGMP 测试缺乏组播业务服务器这样的风险，通过从组织层面采购两台服务器，即可以将这样的风险避免；系统需求规格说明中针对 IGMP 容量需求无法进行测试的风险，通过组织测试人员开发模拟器，模拟大量用户加入和退出组播业务，从而避免容量可测试性方面的风险。

3）风险转移

将风险转移到其他方，例如，买保险就是一种风险转移。又例如，软件产品的一个功能模块，开发团队没有相关的开发经验，为了保证质量和进度，将该功能模块的开发外包给了外包方，这实际上就采取了风险转移的策略。当然，风险转移的过程中通常会产生新的风险，例如，和外包公司进行管理和沟通的风险。

4）风险接受

识别的风险经过分析和评估之后，认为风险级别在可以接受的范围之内（例如，在组织定义的风险阈值之内），这个时候就可以接受风险；或者针对风险进行应对的成本，比不进行应对的成本还要高，这个时候也可以考虑接受风险，例如，iBAS R1.0 的 IGMP 实现中有一个扩展性方面的风险，经过分析和评估之后，发现假如针对扩展性风险修改 IGMP 实现架构方案，引入的风险更大，因此只能采取风险接受的策略。假如风险无法采取合适的应对措施和活动来避免、转移或者减轻风险，这个时候也只能采取接受风险的策略。

5）应急计划

当风险真的发生了，即风险的可能性达到了 100%，这个时候需要相应的应急计划，降低该事件产生的影响程度，例如，测试团队中处于测试关键路径的测试人员离职这样的风险，在测试过程中如果真的发生了，测试经理就需要启动应急计划，解决这个问题（例如，从其他测试团队借调测试人员来满足当前测试任务的要求）。

前面分别描述了风险减轻、风险避免、风险转移、风险接受和应急计划等不同的风险

缓解策略，测试过程中具体选择哪种风险缓解策略，需要分析和评估该风险缓解策略可能带来的收益和机会，以及相关的成本和潜在的额外风险。风险缓解策略的选择需要在收益、成本和可能产生的新风险之间进行平衡。

在基于风险的测试策略中，通常会将风险识别、风险分析和风险缓解作为制定主测试计划和其他测试计划的基础和重要输入。每个风险条目的风险级别决定了用于处理该风险的测试工作量。一些安全相关的标准（例如 FAA DO-178B/ED 12B、ISO/IEC 61508）规定了基于风险的测试技术和覆盖率要求。

根据风险发生的可能性和严重程度得到的风险级别，可以作为决定测试优先级的依据。根据风险级别确定测试优先级，可以采用以下两种不同的方法。

（1）深度优先：首先执行高风险的测试，完成所有高风险的测试执行之后，才开始低风险的测试执行，即测试执行顺序严格按照风险级别来制定。

（2）广度优先：测试优先级按照抽样的方式来制定。抽样方法根据不同的风险级别，确定不同的权重，测试样本的选择将基于风险的权重而展开。广度优先得到的测试优先覆盖所有已经识别的风险，因此，可以通过风险覆盖率评估得到的测试样本。

除了基于风险的深度优先和广度优先测试策略之外。测试人员还可以使用其他的风险缓解方法减轻风险，主要包括：

1）评审开发工作产品

开发工作产品生成之后，应该根据不同的要求进行不同形式的评审，避免将工作产品中的缺陷带入下一个阶段，例如，避免将系统需求规格说明中的缺陷带入到系统设计规格说明中。发现和修复缺陷的成本会随着开发进度而增加，同时缺陷具有雪崩现象，即缺陷随着开发阶段推进有放大效应。为了降低成本、规避风险以及提高效率，在前期对工作产品进行评审和检查是很好的风险缓解方法。

2）评审测试工作产品

测试工作产品生成之后，例如，测试设计规格说明、测试用例规格说明，需要系统人员、开发人员等参与评审。通过对测试工作产品的评审，可以减少测试范围、测试覆盖率、测试用例正确性等方面的错误，从而减少由于测试工作产品的问题而导致的风险。例如，检查测试用例中的输入数据和网络拓扑结构是否能够验证系统需求的条目。

3）选择不同的测试级别

除了对工作产品进行评审以外，还可以通过定义不同的测试级别，例如，组件测试、集成测试、系统测试等，利用动态测试的方法减少产品的风险，例如，编码阶段之后对模块进行相应的组件测试，可以降低在编码阶段引入的缺陷遗留到下个阶段的风险。

4）选择测试设计技术

不同的测试设计技术，在发现缺陷和提供信息方面的作用是不一样的，例如，对于复杂而重要的功能模块，建议尽量包括低级别的测试，例如，组件测试。在组件测试中，选择测试强度更高的测试设计技术（例如改进的判定-条件覆盖），尽早发现其中的缺陷，从而降低整体成本和缩短后期的测试时间。

5）选择经验丰富的测试人员

根据风险分析和评估的结果，让经验丰富和技术能力强的测试人员（不管是测试设计人员、测试实施人员还是测试执行人员）负责最重要的模块或测试对象，降低其中的风险，例如，经验丰富的测试人员可以设计更加完善、完备和准确的测试用例，实现高质量的测

试用例脚本和代码，以及更加高效地测试和发现测试对象中的缺陷等。同时，让新员工或者经验比较少的测试人员测试低风险的对象和模块，使他们在测试过程中可以慢慢积累经验，同时使得风险处于可以控制和可接受的范围之内。

6）执行确认测试

软件或者软件产品中的缺陷修复之后，需要对这部分修改进行确认测试，以确认缺陷是否正确地修复了。执行确认测试可以降低缺陷没有正确修复的风险。

7）执行回归测试

一般来说，只要测试对象有了变更，就应该进行回归测试。变更可以是用户需求的变化、功能的增加或者删除、缺陷的修复甚至可以是软件系统运行环境的变化等，这些变更可能会给软件带来风险，造成软件产品出现新的问题。回归测试的目的是用来确保软件的变更没有影响软件产品以前工作正常的部分（即没有在软件产品中引入新的缺陷）。

4．生命周期中的风险管理

如果测试组织有测试方针或者测试策略文档，在这些文档中应该描述管理产品风险和项目风险的一般过程，以及如何将风险管理集成到测试的每个阶段，并使其发挥作用。

理想情况下，风险管理应该贯穿于整个软件开发生命周期，而不仅仅发生在测试过程。重要的风险不仅在特定测试级别的前期就得到处理，而且在早期的测试级别中也得到了处理。例如，假如软件产品性能是一个关键的产品风险区域，性能测试不仅在系统测试的前期就会开始，而且在组件测试和集成测试时也会运行性能测试。

测试执行之前也可以缓解软件产品风险，例如，风险识别过程中发现了需求相关的问题，风险缓解的手段可以是项目团队实施全面的需求规格说明评审。通过评审不仅可以降低风险，同时也意味着后续可以减少测试用例来缓解剩余的产品风险。

测试执行期间，基于风险的测试可以帮助项目利益干系人根据剩余风险级别监控软件开发生命周期，包括做出是否发布软件产品的决策。这就要求测试经理应该以剩余风险级别的方式向管理层报告当前的测试状态，让管理层决定是否需要增加测试时间开展更多的测试，或者将剩余风险转移给用户。

测试团队应该不断收集软件产品风险相关的信息，在软件项目的主要里程碑点不断调整产品风险数据，以此指导测试调整。产品风险调整包括识别新的风险、重新评估已有的风险级别、评估风险缓解活动的有效性等。例如，需求阶段针对需求规格说明开展了风险识别和评估，那么在设计规格说明阶段，应该对需求规格说明的风险重新进行评估。再例如，如果测试过程中发现某个组件包含的缺陷远比预期的缺陷数多，则可得出在此区域存在缺陷的可能性比预期要高的结论，因此需要将风险可能性和整体风险级别上调，根据风险级别调整结果，相应增加该组件测试的工作量。

成熟的组织不仅识别风险，还需要识别风险来源，以及风险发生之后将带来的负面后果。对于那些确实发生的风险，需要开展根本原因分析来深入了解风险的来源以实施过程改进，防患于未然。

2.3.2 基于风险的测试技术

基于风险的测试技术各种各样，有些技术非常正式，而有的相对不正式，它们在文档类型、测试级别和运用形式等方面会有所不同。例如，测试人员在探索性测试过程中分析

产品风险的方法（例如缺陷攻击）。尽管这种方法有助于指导测试，但该方法过分关注产生缺陷的可能性，而弱化了缺陷导致的影响，并且测试过程中没有引入跨职能利益干系人的参与。同时该方法相对主观，更依赖于每个测试人员的经验、技能和偏好。因此，采用该方法只能获取部分基于风险的测试的收益。

为了更好地体现基于风险的测试的优势，测试过程中更多地会采用相对正式的测试技术。根据这些技术的特点，可以将它们划分为两类：轻量级的基于风险的测试技术和重量级的基于风险的测试技术。

1．轻量级的基于风险的测试技术

测试团队采用轻量级的基于风险的测试技术，可以以较小的成本获得基于风险的测试的收益。该方法融合了非正式方法的快速响应能力和灵活性，同时可以获得正式方法的部分力度和一致性。轻量级基于风险的测试技术包括：

（1）实用风险分析与管理（PRAM）；

（2）系统的软件测试（SST）；

（3）产品风险管理（PRisMa）。

轻量级的基于风险的测试技术，除了基于风险测试的常见属性外，这些技术通常还有以下特点。

（1）从基于风险的测试行业经验中逐渐演变而来，特别是那些效率至关重要的行业。

（2）其有效性取决于跨职能部门的不同利益干系人在初期风险识别和评估中的广泛参与，例如，来自不同业务的代表和具有不同技术观点的干系人。

（3）其有效性可以通过以下方式进行优化：软件项目的早期阶段就引入可最大化缓解产品风险的选项，以及确认风险识别的主要工作产品和副产品，有助于影响软件产品的规格说明和软件产品实现的风险最小化。

（4）风险管理过程中生成的输出结果，例如，风险矩阵或者风险分析表，作为测试计划和测试条件的组成部分，以及后续测试评估和管理的基础。

（5）对剩余风险进行评估，并汇报给各级别的利益干系人。

不同轻量级的基于风险的测试技术，相互之间有不同的特点。例如，系统的软件测试 SST，要求将软件产品的需求规格说明作为风险分析的输入，而且只有在提供了需求规格说明之后才能使用。而实用风险分析与管理 PRAM 和产品风险管理 PRisMa，则鼓励采用基于风险与基于需求混合的策略，除了需求规格说明或者其他规格说明作为风险分析的输入之外，也可以完全基于利益干系人提供的输入（例如：与用户进行面对面的沟通）作为分析的基础。

将需求作为风险分析的输入，可以确保对需求的覆盖率，但测试经理需确保不漏测在需求中没有提到的重要风险，尤其是非功能领域的风险。如果得到的输入是完整、高质量且设定了优先级的需求，通常情况下测试团队可以更加容易实现需求和风险之间的可追溯性，以及风险级别和需求优先级之间的密切关联。

轻量级的基于风险的测试技术也鼓励测试人员采用风险识别和评估的过程，作为不同利益干系人之间针对测试方法达成一致的一种方式。这种方式可以带来巨大的好处，但同时也会占用利益干系人的时间去参加小组的头脑风暴会议，或者面对面的访谈。假如测试过程没有广泛的利益干系人参与，会导致风险分析过程中出现遗漏。当然，利益干系人并

不一定对每个风险级别意见保持一致，此时产品风险分析的主持人必须能够积极且创造性地协调不同干系人的意见，尽最大可能使他们达成一致。主持产品风险分析的人员所需的技能要求和专门主持评审会议的主持人的技能要求是一样的。

类似更正式的技术，轻量级的基于风险的测试技术允许对风险的可能性和严重程度的影响因素进行加权考虑，以强调业务风险或者技术风险。与更正式的技术相比，轻量级技术有下面两个特点。

（1）仅考虑两个因素：风险的可能性和严重程度。

（2）采用简单的定性方式评估风险级别，例如，高、中、低。

轻量级的基于风险的测试技术不仅可以在诸多领域带来灵活性和可用性，同时也适合整个团队不同经验和技能水平的人员应用，甚至是非技术人员或者新人应用。

2．重量级的基于风险的测试技术

除了上述轻量级的基于风险的测试技术，测试经理还可以选择重量级的测试技术，主要的选项包括：

（1）危害分析（Hazard Analysis），其目的是识别过程危害，估算它们的风险，并决定风险是否可以接受。主要的任务包括危害识别、分析危害可能性和严重程度。危害分析的特点是将风险分析延伸至整个分析过程的上游，尝试识别每个风险背后隐藏的危害。

（2）暴露的成本（Cost of Exposure），其主要特点是针对每个产品风险，风险评估过程包括三个因素：①与该风险项相关的失效的可能性（用百分比表示）；②一旦过程中发生与该风险项相关的典型失效将造成的失效成本（用成本单位表示）；③发现这些失效的测试成本；

（3）失效模式和影响分析（Failure Mode and Effect Analysis，FMEA）和其他变体版本[Stamatis03]，其主要特点是首先识别软件的产品风险，产生风险的潜在原因和可能造成的影响，然后确定严重程度和检测难度等级，再计算优先级因子，并基于优先级因子确定风险处理的优先级。

（4）质量功能展开（Quality Function Deployment，QFD），属于产品风险管理技术。其主要特点是：关注由于对客户或用户需求理解不当或不充分而引起的产品风险。

（5）故障树分析（Fault Tree Analysis，FTA），对测试中实际发现的各种失效，不管是测试过程还是生产过程，或者潜在失效（例如产品风险），都需要对引入的缺陷进行根本原因分析。其基本步骤是：首先分析哪些缺陷会导致软件产品出现失效；其次不断分析哪些错误或者缺陷可能会导致软件产品出现缺陷，直到识别出出现失效的根本原因。

测试过程中具体采用哪种基于风险的测试技术，依赖于软件项目、开发模型和软件产品等的特点。例如，类似 Whittaker 的探索性测试中的非正式方法，可能适用于软件版本小的维护补丁或者针对缺陷的快速修复之后的回归测试。在敏捷开发模型中，产品风险分析会集成到每个迭代（Sprint）的早期，并且将风险分析结果的文档也融入到用户故事中，以方便跟踪和监控。综合系统不仅要求对单个系统进行风险分析，对整个综合系统也有同样的要求。安全关键系统和核心关键项目，对风险分析的正式程度和文档级别的要求会相对更高。

不同基于风险的测试技术，其输入、过程和输出都会有所不同。常见的输入包括：利益干系人对软件产品的见解、规格说明和历史经验和数据等。常见的过程包括：风险识别、风险评估和风险控制。常见的输出包括：产品风险列表（附带有其他信息：各个风险的风

险级别、建议的测试工作量分配等)、测试依据等输入参考文档中发现的缺陷列表、风险项相关的疑问或者问题列表，以及影响测试或者项目进度的项目风险。

邀请广泛的项目利益干系人积极参与风险识别和评估，是成功实施基于风险的测试的关键因素。根据利益干系人对产品质量理解的不同，可以将他们划分成两大类：业务干系人和技术干系人。他们对产品风险的识别和评估有不同的关注点和优先级确定方式。

（1）业务干系人包括客户、用户、运营人员和技术支持人员等。这些干系人了解客户和用户如何使用软件产品，因此可以从业务的角度识别风险和评估风险的严重程度。

（2）技术干系人包括开发人员、架构师、数据库管理员和网络管理员和测试人员等。这些干系人知道潜在的软件失效方式，因此可以从技术层面识别风险和评估风险的可能性。

除了单纯的业务干系人和技术干系人，有的干系人兼具业务和技术能力，因此他们既可以代表业务角度的观点，也可以代表技术角度的观点。例如，担任测试分析或业务分析角色的专题领域的事务专家，由于他们在技术和业务领域方面都具备专业知识，因此他们一般会有更广阔的视角，可同时进行风险识别、评估风险可能性和严重程度。

识别风险的过程会输出一张风险列表，只要利益干系人认为软件产品中存在产品风险，那它们就可以认为是风险，没有必要争论风险项的具体内容，更重要的是利益干系人对风险级别能够达成共识。例如，轻量级的基于风险的测试技术中，采用了可能性和严重程度作为评估的两个因素，因此干系人必须确定可能性和严重程度的量化方案。同时要求测试团队的所有干系人也必须用同样的量化方案，且必须能覆盖每个产品风险的可能性和严重程度。

测试经理必须让项目利益干系人看到基于风险的测试技术的价值，理解干系人的要求、期望和参与过程的可用时间等，因为干系人积极参与风险分析过程对测试经理而言是至关重要的。即使在软件产品需求缺失或者描述不明确的情况下，干系人仍然可以在合适检查表的引导下识别产品风险。对测试而言，实施了基于风险的测试之后，可以提高缺陷检测的效率和有效性，开展更加全面的测试，同时会为干系人带来各方面的收益。

完成基于风险的测试之后，测试团队需要衡量他们的收益。下面 4 个问题可以帮助判断测试团队是否达到了目标。

（1）测试团队发现的重要缺陷是否比次要缺陷要多？

（2）大部分的重要缺陷是否是测试团队在测试执行的初期发现的？

（3）测试团队能否就风险向干系人解释测试结果？

（4）测试团队没有执行的测试用例（如果有的话）是否比已经执行的测试用例的风险级别更低？

假如上面 4 个问题的答案是肯定的，通常情况下可以说明基于风险的测试实施是成功的。除了完成基于风险的测试，测试经理还应该针对上述度量设定过程改进目标。测试过程中涉及的度量有很多，测试经理应该仔细考虑这些度量之间的关系，以及管理这些度量所采取的测试活动。

2.3.3 测试用例选择的其他技术

基于风险的测试技术是测试经理选择测试用例和安排测试优先级的最常用的一个策略，此外，测试经理在进行测试用例选择时，还会选择一些其他技术作为辅助，主要包括

以下几种。

1．基于需求的测试技术

基于需求的测试技术，是非常流行的开发测试条件并设定测试优先级的一种方式。常见的基于需求的测试技术包括歧义评审、测试条件分析和因果图分析。

歧义评审，其主要目的是识别和消除作为测试依据的需求文档中可能存在的歧义。根据评审对象质量、时间、资源等要求的不同，歧义评审可以采用从非常不正式到非常正式的评审过程，例如，非正式评审、走查、技术评审和审查。评审过程中明确评审目标、管理层的支持、具备相关技能的评审人员的参与等，对于提高评审的效率和有效性至关重要。评审过程中评审人员能够得到合适的评审检查表的支持，也可以提高评审过程的有效性，例如，需求评审过程中，参考常见需求缺陷的评审检查表。

测试条件分析，主要指的是测试人员仔细阅读需求规格说明，熟悉软件产品的工作原理和业务流程，主要目的是尽量多地识别测试条件以覆盖需求。假如需求规格说明中的需求已经设定了优先级，那么它们同样可以用来设定测试条件、测试用例和分配测试工作的优先级。假如需求规格说明中的需求没有设定优先级，则需要结合基于风险或其他的测试技术，否则将难以确定合适的测试工作分配和测试执行的先后顺序。

因果图分析，作为高级测试分析师模块中测试技术的一种，其主要目的是分析测试依据，获得测试条件的组合，更好地覆盖测试依据。实际上，因果图分析有更广泛的用途，它可以将庞大的测试问题归纳为数量可控的测试用例，并且仍能保证软件产品功能的 100%覆盖。通过因果图分析还可以在设计测试用例时识别出测试依据中的分歧。如果在需求规格说明初稿阶段就开始设计测试用例，就可以在软件开发生命周期的早期发现缺陷。使用因果图分析的一个难点是画出因果图。假如因果图过于复杂，通过人工方式很难完成，那么需要相应工具的支持。

基于需求的测试技术面临的一个挑战是：需求规格说明模糊不清、可测试性差、需求不完整甚至需求根本不存在。并不是所有的团队都会积极解决这些问题，如果面临这种情况，测试人员必须选择其他测试策略。

2．基于运行概况的测试技术

基于运行概况的测试技术，是另一种会用到现有需求规格说明的方法。该方法通过创建概况或者运行概况，综合用例、用户（有时称作参与者）、输入和输出等构件模型，以准确地展现该软件系统在现实中的应用。通过这样的方式不仅测试了软件系统的功能性，同时还可以测试软件对象的易用性、互操作性、可靠性、安全性和性能等非功能特性。

在测试分析与计划期间，测试团队识别运行概况，并尝试设计测试用例覆盖这些概况。运行概况只能是基于当前可用的信息，对实际软件系统的一种估算。也就是说，和基于风险的测试一样，运行概况也可能无法完全展现软件系统的真实情况。不过，如果有充足的信息和干系人输入的话，那么通过运行概况构建的模型，可以很好地展现软件系统的实际情况。

3．基于检查表的测试技术

基于检查表的测试技术，也是选择测试的一种手段。测试经理通过检查表的帮助，可以有条理地决定要测试什么，测试多少，以及按照怎样的顺序进行测试。假如软件产品已经比较稳定，那么列出所有待测的主要功能和非功能区域的检查表，加上已有的测试用例，就可以保证测试比较充分。

测试过程中采用的检查表，假如基于变更类型和数量为基础，通常可以对测试分工和测试顺序的安排提供启发。但这种方法只在变更范围比较小的情况下，测试才会比较有效。

4．应对型的测试技术

最后，测试团队另一种常用的方法是应对型的测试技术。该测试技术的特点是，测试执行之前的测试分析、测试设计和实施等活动做得比较少。测试团队更关注的是应对实际交付的软件产品。假如在测试过程中发现了缺陷集群，该缺陷集群就成了后续测试的重点。测试优先级的设定和分配，都是动态调整的。应对型的测试技术应该作为其他方法的补充。因为如果单独采用该技术，往往容易遗漏掉软件产品中的一些至关重要的区域，例如，仅仅因为在某些功能区域没有发现大量的缺陷遗漏一些基本功能的测试。

上述的测试选择技术并不是独立存在的，测试过程中应该是相互支持的，结合前面讲的基于风险的测试技术，才能在测试过程中尽早尽量全面地识别和选择测试用例，并对测试用例合理地安排测试执行的优先级。

2.3.4　测试过程中的测试优先级设定和工作量分配

前面阐述了测试优先级设定和工作量分配的测试技术，测试经理不管选择采用哪种技术，或者混合使用一些技术，测试经理都必须将该技术融入软件项目以及测试过程中。例如，在顺序模型（例如 V 模型）中，测试团队在需求阶段识别测试条件、分配测试工作量，并开始设定测试的优先级，然后再进行定期调整；而在增量-迭代或者敏捷开发生命中，要求采用逐步迭代的方式识别和调整测试用例，测试计划和控制必须考虑到风险、需求和运行概况，并根据测试过程中收集的信息做出相应的应对。

在测试分析、设计和实施期间，必须依据测试计划阶段做的测试分工和优先级开展测试工作。假如这些信息没有用来指导后续的测试过程，往往会导致测试分析或者测试建模碰到困难，直接影响后续测试活动的质量。

在测试执行期间，测试执行的优先级应该参考测试计划阶段设定的测试优先级。不过测试经理需要关注的一点是，测试计划阶段设定的测试优先级，应该根据不同阶段获取的信息，定期进行更新，例如，每个阶段里程碑点时。在评估测试出口准则和汇报测试结果时，测试经理必须考虑风险、需求、运行概况、检查表，或者其他用于测试选择和优先级设定的信息。假如有必要，测试经理需要根据测试优先级方案对测试进行分类。

作为测试结果汇报和出口准则评估的一部分，测试经理需要度量测试完成的程度，包括将测试用例和发现的缺陷，与对应的测试依据之间保持可追溯性。例如，在基于风险的测试过程中，随着测试执行不断发现和修复缺陷，测试人员可以评估软件产品的剩余风险级别。通过基于风险的测试过程中得到的信息，帮助管理团队确定软件版本发布的时机。测试汇报中应该提到已经覆盖的风险和还处于未关闭的风险信息，以及已经获取的收益。

最后，在测试结束阶段，测试经理根据客户和用户的质量要求，评估测试成功准则。并对测试优先级设置和调整、测试工作量的分配和变更等方面进行经验教训总结，以改进后续软件项目的效率和有效性。只有测试团队满足不同干系人的需求和期望结果时，测试团队才可以称得上是真正有效的团队。

2.4 测试文档和其他工作产品

软件测试过程由不同的测试阶段组成，每个阶段都会有相应的测试工作产品的输出。测试过程的每个阶段编写和输出测试文档是测试管理的重要组成部分。测试管理文档的名称和范围在不同组织和项目中会有所不同。以下是测试管理文档的通用类型。

（1）测试方针：描述组织关于测试原则、方法和主要目标的高级文档。测试方针主要包括组织针对测试的定义、测试过程定义、测试的评估以及测试过程改进的方法等。

（2）测试策略（测试指南）：它是一个高级文档，该文档主要包括需要对软件产品或者程序（一个或多个项目）执行的测试级别和需要进行的测试的定义和描述。测试策略描述了组织的测试方法，包括产品风险和项目风险管理，将测试细分为步骤、级别或阶段，以及和测试相关的概要活动。

（3）主测试计划（项目测试计划、测试方法）：通常针对多个测试级别的测试计划。它描述了将测试策略应用于一个特定项目，包括项目中需要开展的各种测试级别，以及这些测试级别之间的关系。

（4）级别测试计划（或阶段测试计划）：通常用于一个测试级别的测试计划。它描述在某个测试级别中所要进行的特定活动，包括某个特定测试级别对主测试计划的拓展。

不同的组合和项目，采用的项目类型和名称是不同的。有些组织和项目可能会将上面这些不同类型的文档合并成一个文档，文档内容也可能在其他文档中出现。文档中的某些内容也可能是凭直觉而以非书面的形式表现出来，或者以传统的测试方法表现出来。较大且较正式的组织和项目倾向于撰写上述所有类型的文档，并将其作为书面的工作产品予以保存，而规模较小的组织和项目通常只会撰写其中的一部分。

本节将分别描述上面提到的测试管理文档，同时还会对 IEEE Std 829—2008 中涉及的其他相关文档的内容进行简单的描述（参考 2.4.6 节的内容）。图 2-3 是本节涉及的测试文档关系图。

2.4.1 测试方针

测试方针描述了组织或者项目的测试原则、测试方法和主要测试目标。测试方针以文档形式或者作为一种管理理念而存在，它规划了组织或者项目期望达到的总体测试目标。测试方针可以由组织的信息技术部门、研发部门或产品开发部门制定。不管由谁负责制定，测试方针都应该反映组织或者项目在测试方面的价值和目标。

测试方针有时也可以作为质量方针的一部分或有效的补充，质量方针描述了质量管理的整体价值和目标。测试方针是一篇简短而概要的文档，主要包含下面这些内容。

（1）明确测试的目的，例如，提供软件系统按照预期运行的信心和检测缺陷。

（2）定义基本的测试过程，例如，测试计划和控制、测试分析和设计、测试实施和执行、评估出口准则和报告以及测试结束活动。

（3）描述如何评估测试的有效性和效率，例如，缺陷发现百分比、测试阶段发现缺陷

和产品发布后发现缺陷的修正成本的比较。

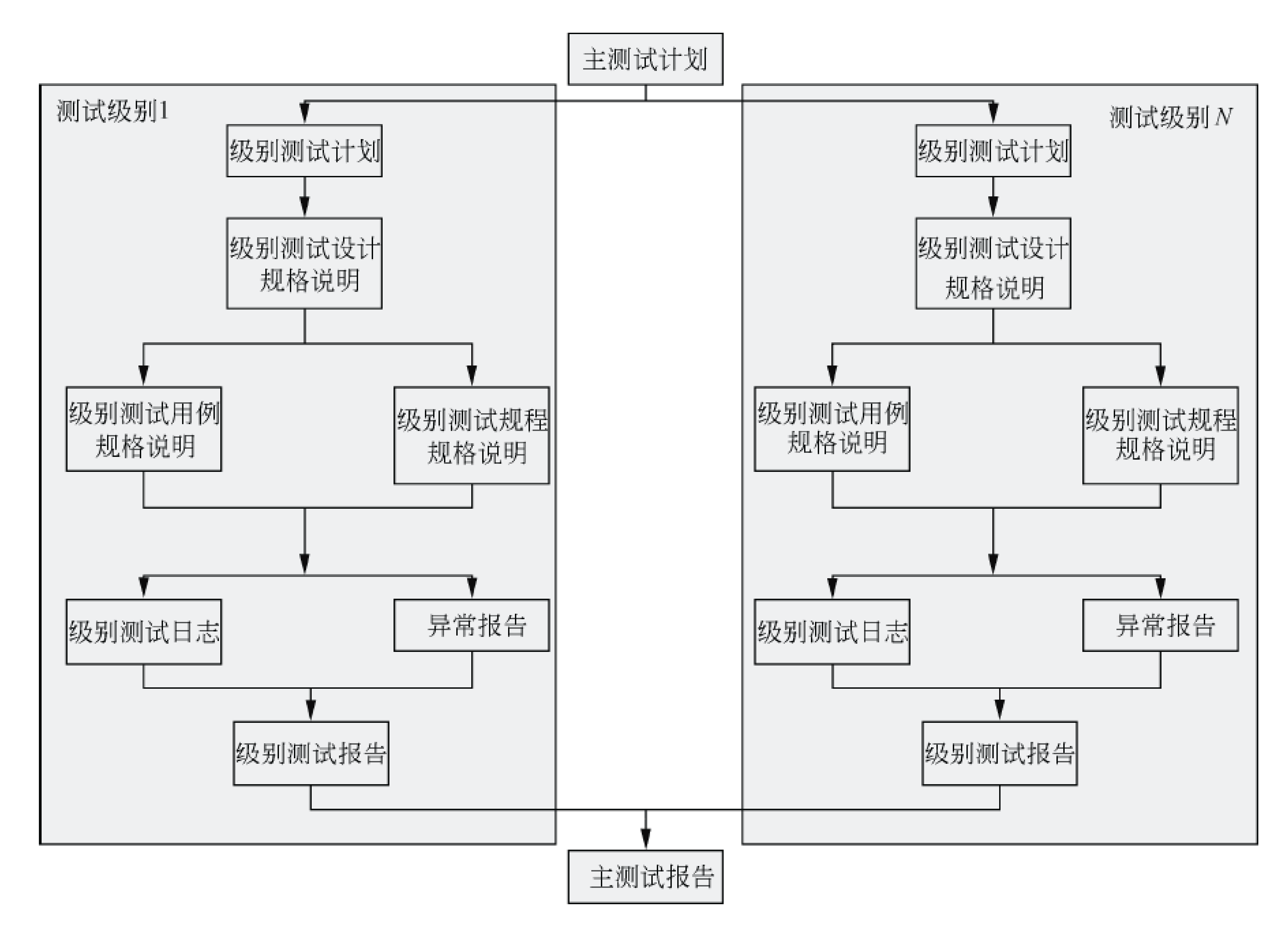


图 2-3　测试文档关系图

（4）定义期望的质量目标，例如，可靠性（可通过失效率来度量）或易用性要求。

（5）规定测试过程改进的活动，例如，应用测试成熟度模型集成（TMMi）、测试过程改进模型（TPI-Next），或根据过去的项目经验改进测试过程。

测试方针既可以应用于新开发的软件项目，也可以针对维护项目。测试方针需要参考组织层面定义和使用的测试术语标准。

2.4.2　测试策略

测试策略描述了组织或项目采用的测试方法，包括产品项目和项目风险管理、测试级别和阶段定义，以及测试的概要活动等。测试策略中描述的测试过程和测试活动应与测试方针的定义内容保持一致。测试策略包括针对组织或针对多个项目所需的通用测试要求，例如，通用测试入口准则和出口准则。

基于测试活动开始时间的不同，测试策略可以分为两类：预防型策略（尽早设计测试以预防缺陷）和应对型策略（测试活动在软件或系统实现后开始）。典型的测试策略如下。

1．分析式的测试策略

常见的分析式的测试策略有：基于风险的测试、基于需求的测试。该测试策略下，测试团队通过分析测试依据，尽量多地识别测试条件。例如，在基于需求的测试中，通过测试分析从需求中导出测试条件，再设计和实施测试用例来覆盖测试条件。然后执行测试用例，其执行的先后顺序通常参考对应需求的优先级。测试进度监控和评估贯穿于整个测试过程，测试结果的汇报主要基于测试覆盖的需求状态，如测试已经通过的需求，测试未通

过的需求，还未完全测试的需求，测试受阻的需求等。

2. 基于模型的测试策略

基于模型的测试策略，例如，基于运行概况的测试。该测试策略下，测试团队开发一个模型，该模型展现了软件系统的运行环境（可以是基于实际或期望的情况），系统应该接受的输入和系统运行的条件，以及系统应如何运行和为用户提供什么样的服务。例如，在针对高速发展的移动设备应用程序的基于模型的性能测试中，基于当前的使用情况和项目随着时间推移的发展情况，需要开发出展现未来的网络通信量、活跃和不活跃的用户以及造成的处理负载的模型。另外，也可以开发出当前生产环境的硬件、软件、数据容量、网络和基础设施的模型。还可能开发出理想的、预期的和最小的吞吐率、响应时间和资源分配的模型。

3. 基于方法的测试策略

基于方法的测试策略，例如，基于质量特性的测试。在该测试策略下，测试团队使用预先定义的测试条件，例如，ISO/IEC 9126 质量标准（ISO/IEC 25000 系列将会替代 ISO/IEC 9126 标准），与特定领域、应用或测试类型（如安全性测试）相关的通用化逻辑测试条件的检查列表或集合等，该系列测试条件将会在每个迭代或每次发布期间使用。例如，针对简单稳定的电子商务网站进行维护测试时，测试人员可能会采用已经识别的关键功能、属性和链接的检查列表作为测试的输入。该商务网站每次出现变更时，测试人员都会覆盖检查列表上的相关要素。

4. 基于特定过程或标准的测试策略

基于特定过程或标准的测试策略，例如，受制于美国食品药品管理局（US FDA）标准的医疗系统的测试。在该测试策略下，测试团队遵循由标准委员会或其他专家组制定的一系列过程，该过程描述了需要交付的文档、测试依据、测试准则和测试团队的正确识别、构建和使用。例如，在遵循敏捷管理技术的项目中，测试人员需要在每个迭代期间分析描述软件产品特定特性的用户故事，估算每个特性所需的测试工作量，并将它们作为该迭代计划的组成部分。针对每个用户故事识别测试条件（通常称为验收准则），执行覆盖这些测试条件的测试用例，并在测试执行过程中将每个用户故事的状态（例如，未测试、测试失败或测试通过）汇报给团队负责人。

5. 动态和启发式的测试策略

动态和启发式的测试策略，例如，基于已知缺陷的攻击测试。在该测试策略下，测试团队在开发团队完成软件开发之后才开始测试设计和实施，对待测的实际软件系统做出被动的测试应对活动。例如，在基于菜单的个人计算机应用程序进行探索性测试时，可能会开发出一套与特性、菜单选择和屏幕对应的测试章程。每个测试人员都分到了一套测试章程，然后使用这些章程来组织探索性测试的会话。测试人员定期向测试经理汇报测试会话的结果，测试经理再根据发现的结果修改测试章程。

6. 咨询式的测试策略

咨询式的测试策略，例如，用户导向的测试。在该测试策略下，测试团队根据一个或多个关键利益干系人的输入决定需要覆盖的测试条件。例如，在对基于网站的应用程序进行外包兼容测试时，公司可能会提供外包的测试服务供应商一份浏览器版本、反病毒软件、操作系统、连接类型和其他配置项的优先级清单，并且将以这份清单作为对该应用软件进

行评估的基础。然后测试服务供应商就可以用诸如两两组合测试（针对高优先级测试项）的测试技术和等价类划分（针对低优先级测试项）来生成测试。

7．反回归测试的测试策略

反回归测试的测试策略，例如，采用广泛的自动化进行回归测试。在该测试策略下，测试团队使用各种技术来管理回归测试风险，尤其是针对多个测试级别的功能或非功能回归测试自动化。例如，如果对一个基于网站的应用程序进行回归测试，测试人员可以使用基于图形化用户界面 GUI 的测试自动化工具来自动化该应用的通用和特殊用例。然后在每次修改该应用程序时执行这些测试。

前面简单描述了典型的测试策略，这些策略并不是孤立存在的，在测试过程中应该根据实际情况组合应用不同的测试策略，例如，在基于质量特性的基础上，采用基于风险的测试。组织选定的测试策略应满足产品特点和组织要求，并可根据其特定的业务或项目特点进行合理的裁减。

假如测试策略中描述了产品风险和项目风险，以及在测试过程中如何管理这些风险，那么需要对风险和测试之间的关系进行解释，并说明风险应对和风险管理的可选方案。

测试策略中还需要描述测试过程中包含的测试级别。在这种情况下，应给出制定每个测试级别的入口准则和出口准则的概要指导，以及不同测试级别之间的关系（例如，不同的测试级别对应不同的测试覆盖目标）。

除了上述提到的内容，测试策略也可包括以下方面的内容。

（1）集成规程；

（2）测试规格说明技术；

（3）测试独立性（可根据不同的测试级别而有所变化）；

（4）必需的和可选的应遵守的标准；

（5）测试环境；

（6）测试自动化；

（7）软件工作产品和测试工作产品的可重用性；

（8）再测试和回归测试；

（9）测试控制和测试报告；

（10）测度和度量标准；

（11）事件管理；

（12）测试件的配置管理方法；

（13）角色和职责。

测试策略既可以是组织内针对测试的短期策略，也可以是长期的策略。但是不管如何，测试策略都应该以文档的形式体现出来。同时，测试策略的制定依赖于组织架构、产品类型、开发模型、成员技能等特点，不同组织和项目需要不同的测试策略来满足要求，例如，安全关键的软件应用系统，应使用比其他应用系统更完整和严格的测试策略。

2.4.3　主测试计划

主测试计划的目的是对多个测试级别提供一个整体的测试计划和管理文档。主测试计

划包含测试工作量的选择、测试目的设置、测试资源的设置（时间、设备资源等）、阐述测试对象之间的关系、识别风险和假设条件、测试参考的标准和规范、测试工作量控制、对质量保证计划中质量的承诺等内容。同时，主测试计划也定义了不同测试级别计划和测试级别之间的关系、整体的测试任务和测试文档的需求等。

图 2-4 是 IEEE Std 829—2008 中定义的主测试计划的大纲目录，它主要由三大部分组成：介绍、详细内容和其他一些需要规定和定义的内容。

主测试计划大纲

1. 介绍
   1.1 文档标识
   1.2 范围
   1.3 参考资料
   1.4 系统概述和主要功能
   1.5 测试概述
      1.5.1 组织结构
      1.5.2 主测试进度
      1.5.3 完整性级别
      1.5.4 测试资源
      1.5.5 角色和职责
      1.5.6 工具、技术、方法和度量
2. 详细内容
   2.1 测试过程定义
   2.2 测试文档需求
   2.3 测试管理需求
   2.4 测试报告需求
3. 其他
   3.1 术语和缩略语
   3.2 文档修改记录

图 2-4　主测试计划大纲目录

1. 介绍

主测试计划的介绍部分描述了它在整个项目开发生命周期中的位置，对项目的测试工作进行整体介绍。它主要包括文档标识、范围、参考资料、系统概述和主要功能、测试概述等内容，同时组织架构、主测试进度和完整性级别、测试所需的资源、角色和职责、工具和技术等也可以在这部分进行介绍。

1）文档标识

文档标识用来唯一地标识主测试计划文档，可以包括文档发布日期、发布组织、作者、批准者和文档的版本状态（如初稿、已评审、已修改或者定稿）等。该标识通常在文档的首页，例如，在封面或者紧跟在封面之后。具体可以根据不同组织和项目的定义来确定文档标识的内容和位置。

2）范围

它主要描述了测试工作的目的、目标和测试范围。本部分可以根据项目的需要，对采用的测试过程和标准进行适当的裁减。该部分阐述了主测试计划针对的项目和测试工作需要覆盖的特定过程和产品，同时，它也描述了测试工作包含的内容、假定条件和限制条件。针对任何测试计划，清楚地定义测试工作的限制条件都是很重要的。

主测试计划中既要定义需要测试的对象，也要包括不需要测试的对象，例如，本次测试只针对该产品新开发的功能，而对于前面版本的旧的功能的测试并不需要在本计划中覆

盖。另外，测试工作中其他因素也会影响测试活动，例如，产品中的部分组件是外部采购的，那么在主测试计划中会假定这部分功能的组件测试和集成测试会由外部厂商自己完成，因此，主测试计划针对这部分组件，将重点考虑该组件和整个系统的集成测试。

测试任务会反映整体的测试方法和开发方法。假如软件开发生命周期模型是瀑布模型，那么每个测试级别都只是执行一次；假如开发生命周期模型是增量迭代模型，那么针对每个测试级别都可能有多个迭代，例如，组件测试可能基于最新的迭代开展，而验收测试可能会基于更早的一个迭代产品来开展。

测试方法确定了各个测试级别的测试内容和执行顺序，同时也描述了和开发方法之间的关系。测试方法也可能描述在不同测试级别中采用的不同的测试方法，例如，在系统测试级别采用基于规格说明的测试技术。采用的测试方法不同，测试过程中需要输出的测试文档也可能是不同的。

3）参考资料

参考资料部分罗列了主测试计划涉及的参考文档。通常来说，参考资料会分为外部参考资料和内部参考资料。参考资料部分在主测试计划文档中的位置既可以在文档的前面，也可以放在文档后面。

（1）外部参考资料：法律、政府法规、标准、政策等。

（2）内部参考资料：项目计划、质量保证计划、配置管理计划、用户需求规格说明等。

4）系统概述和主要功能

该部分描述了被测系统或者产品的商业目的，通常可以从系统文档中找到这方面的信息（例如系统可行性研究报告）。同时，该部分还描述了被测系统或者产品的主要功能。

5）测试概述

测试概述部分描述了开展测试工作需要涉及的组织结构、主测试进度、完整性级别、测试资源、角色和职责、工具、技术和方法等。它们的含义如下。

（1）组织结构：描述了测试过程和其他测试过程之间的关系，例如，开发过程、项目管理、质量保证、配置管理等之间的关系。组织结构也包括测试团队之间的沟通方式、测试任务过程中的授权问题、测试产品和测试过程的批准权限问题等。它可以用可视化的组织结构图进行表述。

（2）主测试进度：描述了在项目开发生命周期和各个里程碑间的测试活动，包括测试任务的整体时间进度、将测试任务结果反馈给开发过程、组织过程和支持过程（例如质量保证和配置管理）的方式。描述了针对测试任务重新执行的任务迭代策略和相互依赖性。

（3）测试级别的完整性：描述了针对软件产品或者基于软件系统的测试级别完整性要求，例如，IEEE Std 829—2008 中定义了 4 个测试级别，从组件测试、集成测试、系统测试到验收测试。针对整个软件产品或系统，不同的测试级别需要完成的测试文档各不相同（测试级别越低，需要完成的测试文档越少）。同时，主测试计划中需要指定针对每个组件的完整性级别，例如，需求、功能、软件模块、子系统、非功能特征等。组件的完整性级别是根据组件中可能发现的缺陷的严重程度来决定的，如果该组件中发现的缺陷可能导致人员死亡或受到严重伤害、设备丢失数据或严重损坏等，那么该组件的完整性级别应该设为最高。完整性级别需要根据变更情况不断进行重评估，例如，由于架构选择、设计选择、代码结构或者其他开发活动引起的完整性级别的变更。

（4）测试资源：描述了测试过程中所需的资源，包括人力资源、设备、工具和其他特殊的需求，例如，安全性需求、接入权限需求和文档控制需求等。

（5）角色和职责：提供测试任务对应的角色和职责。识别组织需要提供的成员和相关的测试职责。

（6）工具、技术、方法和度量：描述了在测试过程中需要使用的硬件和软件、测试工具、技术和方法、测试环境等。描述用来识别和获取可重用的测试件的技术。文档化测试工作需要使用的度量，描述这些度量是如何支持测试目标的。

2. 详细内容

1）测试过程定义

测试过程定义中需要识别和文档化测试过程中需要执行的测试活动和测试任务，它提供了整个软件开发生命周期中的测试活动和测试任务的概述。测试过程定义中应该识别要执行的测试级别以及它们的执行顺序。集成测试通常是在一系列更低级别的测试完成之后进行的，例如，组件集成测试在一系列的组件测试完成之后开展。其中也可能需要定义一些特殊的测试类型，例如，安全性测试、易用性测试、性能测试、压力测试、备份/恢复测试和回归测试等。

对于规模较小的系统，需要的测试级别可能比较少，例如，只进行系统测试和验收测试。假如组织内已经定义了测试过程，这部分内容可以直接引用具体的参考文档。

测试过程中需要详细定义每个测试活动。针对每个测试活动，需要阐述以下 8 个方面的内容。

（1）测试任务：测试活动中需要执行的测试任务。

（2）测试方法：描述针对每个测试任务需要采取的测试方法、步骤，以及工具，并定义用于评价测试任务结果的准则。

（3）测试输入：描述测试任务所需的输入，并描述每个输入的来源。

（4）测试输出：从测试任务中识别所需的输出。当前测试任务和测试活动的输出可能会成为下一个过程或者测试活动的输入。

（5）时间进度：描述了针对测试任务的时间进度。为每个测试任务开始和结束、检查输入和提交输出建立特定的里程碑点。

（6）测试资源：识别开展测试任务所需的资源。根据不同的分类描述资源，例如，人力资源、工具、设备、培训等。

（7）风险和假设条件：识别和测试任务相关的风险和假设条件，并提供建议和应对措施以消除和降低风险。

（8）角色和职责：识别针对每个测试任务需要参与的人员及其各自的职责。

☆示例：编写计划内“测试设计规格说明”任务

表 2-8 是针对测试活动中的编写系统测试设计规格说明任务的一个案例。

表 2-8 测试活动案例

测试任务	编写系统测试设计规格说明
测试方法	按照系统测试计划的要求，生成系统测试设计规格说明，并保证系统测试设计规格说明的目的、格式和内容与 IEEE Std 829—2008 中的测试设计规格说明内容保持一致

续表

测试任务	编写系统测试设计规格说明
测试输入	系统测试计划、系统需求规格说明、IEEE Std 829—2008 中的测试设计规格说明等
测试输出	系统测试设计规格说明
时间进度	项目开始后 30 天内开始任务，项目开始 120 天内完成系统测试设计规格说明，并获得批准
测试资源	参考主测试计划的 1.5.4 节
风险和假设条件	风险：人力资源不能按时到位 假设：测试计划能够按时完成、系统需求规格说明可以按时基线化
角色和职责	参考主测试计划 1.5.5 节

2）测试文档需求

它定义了测试过程中输出的所有测试文档的目的、格式和内容，例如，按照 IEEE Std 829—2008 的格式和内容输出测试过程中的测试设计规格说明、测试用例规格说明、测试规程规格说明等文档。

3）测试管理需求

它描述了异常（例如缺陷）处理和报告过程、任务迭代方针、偏差管理方针、控制规程以及标准、实践和协定等。这些在测试执行过程中是必需的，具体描述如下。

（1）异常处理和报告过程：描述了异常报告和处理的过程，包括报告异常的标准、异常报告的分发、异常处理的授权和时限等。这部分也可以定义异常的严重程度。

（2）任务迭代方针：定义了在输入发生变化或者任务规程发生变化的时候，决定对测试任务进行迭代的准则。这些准则包括对变更、完整性级别、预算、进度和质量等的评估结果。

（3）偏差管理方针：主测试计划和级别测试计划制定之后，需要阐述相关的规程和标准来管理与计划之间的偏差。相关的偏差信息包括任务识别、基本原理、系统和软件质量的影响。同时需要明确批准偏差的责任人。

（4）控制规程：识别适用于测试活动的控制规程。这些规程描述了如何配置、保护和储存基于软件的系统和软件产品。假如质量保证、配置管理、数据管理等活动并没有其他专门的团队负责，那么这些控制规程需要在此处进行描述。

（5）标准、实践和协定：描述了管理测试任务的标准、实践和协定，包括内部组织标准、实践和方针。

4）测试报告需求

它描述了所有测试报告的目的、内容、格式和时间点。测试报告包括测试日志、异常报告、级别中间测试状态报告、级别测试报告和主测试报告等。

3．其他

1）术语和缩略语

提供主测试计划中所涉及的术语和缩略语及其定义。

2）文档修改记录

描述识别、批准、执行和记录主测试计划相关的变更，以及其修改历史和原因等。

☆示例：智能宽带接入服务器 iBAS R1.0 主测试计划

1 介绍

1.1 文档标识

iBAS R1.0 主测试计划

1.2 范围

本文是针对 iBAS R1.0 项目而制定的主测试计划，用于指导 iBAS R1.0 项目的测试活动。针对 iBAS R1.0 项目的测试包括组件测试、集成测试、系统测试和验收测试 4 个测试级别，涉及测试计划和控制、测试分析和设计、测试实施和执行、测试出口评估报告以及测试结束活动各个阶段。iBAS 整个系列产品采取迭代开发的方式，但是在每个迭代内部采用的开发模型是 V 模型，也就是说，iBAS R1.0 项目中采用的是 V 模型，但是在具体操作中要注意和其他后续迭代项目之间的及时沟通。

维护测试的相关计划不在本文讨论范围。

1.3 参考资料

XXX 公司测试方针

XXX 公司测试策略

iBAS R1.0 项目计划

iBAS R1.0 系统规格说明

iBAS R1.0 配置管理计划

iBAS R1.0 质量保证计划

1.4 系统概述和主要功能

iBAS 是智能宽带接入服务器，作为城域网接入层和边缘汇聚层之间的智能业务网关，对以太网、ADSL 接入用户进行身份认证、授权、计费，对其业务施加控制策略，保证用户业务和运营商网络的安全性，同时提供其他增值业务的宽带城域网解决方案。

（1）iBAS R1.0 版本是 iBAS 产品的第一个版本，主要提供如下功能：提供以太网接入。上行接口：支持 100M/1000M 以太网；下行接口：支持 100Base-T、100Base-FX。

（2）支持 RADIUS 认证、计费，支持基于流量、连接时间、分时段等多种计费方式。

（3）支持静态地址和通过 DHCP 动态获得 IP 地址。

（4）支持 PPPoE 用户接入。

（5）通过 VLAN 划分，实现用户二层隔离，限制 VLAN 下接入用户数量。

（6）为接入用户提供组播业务（IGMP 功能）。

（7）支持基于用户级别的流量控制。

（8）提供标准的 SNMP 接口，支持集中网管系统。

1.5 测试概述

1.5.1 测试组织结构

公司采用的组织结构是矩阵式的管理结构，因此，测试团队将根据产品或项目需要，从资源线召集相关人员，产品或项目结束后回到资源线。

本文只针对 iBAS R1.0 项目，所以，这里主要阐述在 iBAS R1.0 项目中测试组织的情况。iBAS R1.0 项目中主要由 5 个团队组成：项目管理团队、开发团队、测试团队、配置管理团队和质量保证团队。这 5 个团队共同负责所有的项目任务，测试经理向项目经理汇报工作。在 iBAS R1.0 的测试团队内部，由 iBAS R1.0 的测试经理统一负责协调管理各项测试工作。该独立的测试团队主要负责集成测试和系统测试，并为组件测试和验收测试提供必要的帮助；组件测试主要由开发人员来完成，测试团队提供必要的工具支持，并参与相关评审；验收测试由第三方独立的测试机构来进行，测试团队为之提供必要的测试环境和技术支持，并进行评审。

1.5.2 主测试进度

在 iBAS R1.0 中将进行组件测试、集成测试、系统测试和验收测试 4 个级别的测试。根据项目开发计划，相应的主测试进度如表 2-9 所示。

表 2-9　主测试进度

里　程　碑	里程碑目标	日　　期
里程碑 1	所有组件测试完成	2008.08.29
里程碑 2	所有集成测试完成	2008.9.30
里程碑 3	所有系统测试完成	2008.11.28
里程碑 4	所有验收测试完成	2008.12.19

各个测试级别都包括以下测试阶段：测试计划和控制、测试分析和设计、测试实施和执行、评估出口准则和报告以及测试结束。具体活动的详细进度安排请参考各个级别测试计划。

1.5.3 完整性级别

根据 iBAS R1.0 项目的实际情况，本项目的完整性级别定义为 4（最高级）。具体的要求参见公司的测试策略。

1.5.4 测试资源

人力资源如表 2-10 所示。

表 2-10　人力资源

序　号	角　　色	人　　数
1	测试经理	1
2	测试分析师	3
3	测试技术分析师	1
4	测试自动化人员	1
5	测试环境管理员	1
6	测试执行人员	6

1.5.5 角色和职责

测试团队涉及的主要角色包括测试经理、测试分析师、测试技术分析师、测试自动化人员、测试环境管理员和测试执行人员。各个角色具体的职责请参考公司的测试策略。

1.5.6 工具、技术、方法和度量

硬件需求如表 2-11 所示。

表 2-11 硬件需求

序 号	描 述	数 量
1	测试用计算机	15
2	iBAS	5
3	SmartBits 6000	2
4	Smartbits 2000	2
5	ADSL Modem	50
6	IGMP 组播业务服务器	1
7	DHCP 服务器	1

软件需求如表 2-12 所示。

表 2-12 软件需求

序 号	描 述	数 量
1	MS Visual SourceSafe 6.0	15
2	MS Project 2003	1
3	Source Insight 3.0	7
4	Visio 2003	4
5	Rational Test Manager	9
6	Rational ClearQuest	9

整个测试活动需要收集和分析的度量很多，具体的指标有：时间进度偏移、工作量偏差、发布前缺陷发现密度、各个文档评审发现的缺陷率、各个测试活动发现的缺陷率等。详细的度量指标参见 iBAS R1.0 质量保证计划。

2 详细内容

2.1 测试过程定义

整个 iBAS 系列产品采用增量迭代开发模型，但是在每个迭代内部，采用的开发模型是 V 模型，也就是说，iBAS R1.0 项目中采用的是 V 模型。在具体操作中要注意和其他后续迭代之间的及时沟通。iBAS R1.0 的测试过程遵循 V 模型对应的 4 个测试级别，并针对 4 个测试级别（组件测试、集成测试、系统测试和验收测试）分别进行测试计划和控制、测试分析和设计、测试实施和执行、评估测试出口准则和报告以及测试结束活动。对于具体的测试级别和测试过程活动的定义，请参考公司的组织测试策略。

每个测试级别中具体的测试活动相关的内容包括具体的进度、人员安排和风险等，请参考具体的级别测试计划。

由于 iBAS 是一个系统集成产品，涉及从底层微码到上层管理软件等多个功能模块，很多测试都需要一定的硬件支持。并且从各个功能的开发交付时间上来看，越是底层的功能（接近硬件）越是先开发完毕。综合分析 iBAS 项目的实际情况，最后决定 iBAS R1.0 项目总的集成策略是自底向上，即先从硬件和底层微码的集成延伸到协议处理，最后到上层的业务管理等功能。但是由于 iBAS R1.0 功能众多，在同一层次上的功能可以同时进行集成，以节约集成测试时间。例如，DHCP 和 PPPoE 接入，在底层功能集成完毕后，这两个功能可以同时集成。

在对各个功能之间进行集成之前，应该先进行功能内部的集成测试，例如，IGMP功能内部包括多个模块，在和其他功能进行集成之前，应该对内部的几个模块进行集成测试，此时由于各个模块较小，且开发交付时间比较接近，可以采取一次性集成的方式开展内部集成测试。

整个 iBAS R1.0 项目将按照上述的集成策略开展集成测试。具体的集成策略请参考集成测试计划。

2.2 测试文档需求

iBAS R1.0 测试活动要求输出的测试文档主要包括：主测试计划、级别测试计划、级别测试设计规格说明、级别测试用例规格说明、级别测试规程规格说明、级别测试日志、缺陷报告、级别测试状态报告、级别测试报告、主测试报告等。这里的测试级别分别是组件测试、集成测试、系统测试和验收测试。

文档的具体格式和要求参见公司的测试策略。

2.3 测试管理需求

iBAS R1.0 的测试任务分配将主要通过管理工具（Rational Test Manager，RTM）完成。测试经理完成测试任务分派后，相关人员登录 RTM 执行相应的任务。所有的测试执行进度和状态必须在 RTM 中进行标识，以保证测试经理能够及时获得测试执行的状态。

所有的测试交付物都要在项目的配置管理之下，具体的配置管理规则请参考 iBAS R1.0 配置管理计划。

对于测试过程中发现的缺陷，全部要求通过 Rational Clear Quest 工具进行提交、跟踪和管理，以保证管理的统一性。

3 其他

3.1 术语和缩略语（部分）

部分术语和缩略语如表 2-13 所示。

表 2-13　术语和缩略语（部分）

术语或缩略语	说　明
ACL	访问控制列表
CLI	命令行接口
DHCP	动态地址配置协议
GUI	图形化用户接口
IGMP	组播协议
SNMP	简单网络管理接口
VLAN	虚拟局域网
WLAN	无线局域网

3.2 文档修改记录

文档修改记录如表 2-14 所示。

表 2-14　文档修改记录

日　期	修订版本	缺陷 ID	修改章节	文档历史	作　者
2008-6-09	draft1.00			初稿完成	xxx

2.4.4 级别测试计划

级别测试计划描述在某个测试级别中所要进行的特定活动，包括级别测试计划对主测试计划的拓展。级别测试计划包含进度、任务和里程碑等在主测试计划中不一定包含的细节信息。另外，级别测试计划还应包括该级别的测试规格说明应遵守的不同标准和使用的模板。

对于非正式的项目或业务，通常只需要撰写一份级别测试计划作为测试管理文档。在这种情况下，该级别测试计划应该包含在下面 1～3 中所提到的内容。图 2-5 是 IEEE Std 829—2008 定义的级别测试计划大纲内容。

级别测试计划大纲

1. 介绍
   1.1 文档标识
   1.2 范围
   1.3 参考资料
   1.4 测试级别
   1.5 测试分类和测试条件
2. 详细内容
   2.1 测试项和标识
   2.2 测试跟踪矩阵
   2.3 测试项
   2.4 不测试项
   2.5 测试方法
   2.6 测试项通过/失败准则
   2.7 挂起和恢复准则
   2.8 测试交付物
3 测试管理
   3.1 计划的活动、任务和进度
   3.2 测试环境
   3.3 角色和职责
   3.4 测试接口
   3.5 资源分配
   3.6 培训
   3.7 进度、估算和成本
   3.8 风险和意外
4. 其他
   4.1 质量保证规程
   4.2 度量
   4.3 测试覆盖率
   4.4 术语表
   4.5 文档修改记录

图 2-5 级别测试计划大纲

有关级别测试计划的主要内容，读者基本都有所了解，这里就不再全面描述。下面对级别测试计划中的部分内容进行介绍，并结合 iBAS R1.0 给出具体的实例（针对 iBAS R1.0 系统测试级别的测试计划）。由于 iBAS R1.0 系统比较庞大，整个系统包括的功能很多，各个功能的测试方法各有不同。为了更有利于读者对级别测试计划的理解，下面例子中的系统测试计划主要关注系统中的 IGMP 功能，其系统测试计划文档名称为：iBAS R1.0 IGMP 系统测试计划。

1．测试项和不测试项

在级别测试计划中，需要明确针对该测试级别覆盖的测试项和没有覆盖的测试项（简称为不测试项）。测试项指的是需要被测试的单个要素，通常，一个测试对象包含多个测试项。测试对象指的是需要测试的组件或系统。

☆示例：iBAS R1.0 IGMP 系统测试计划中对测试项和不测试项定义

在 iBAS R1.0 IGMP 的系统测试计划中，从系统的管理接口层面定义了系统测试覆盖的测试项。iBAS R1.0 为客户提供了三种不同的对系统管理的接口，分别是 CLI（命令行接口）、SNMP（简单网络管理接口）和 GUI（图形化用户接口）。下面是 IGMP 系统测试计划中定义的系统测试中要求覆盖的接口类型。

本文档是针对 iBAS R1.0 项目系统测试级别的 IGMP 功能的测试计划。IGMP 的系统测试主要是对 IGMP 需求规格说明中包括的所有的 IGMP 的需求进行测试。详细的需求条目参见附录 A 中的 IGMP 需求规格说明。iBAS R1.0 为客户提供了三种不同的对系统管理的接口，分别是 CLI（命令行接口）、SNMP（简单网络管理接口）和 GUI（图形化用户接口）。本次测试只覆盖其中的 CLI 和 GUI 两种管理接口。对于 SNMP 管理接口的测试，将有网管团队负责，不在本次系统测试的范围内。

由于 IGMP 的协议处理部分来自于公司开发的内部协议平台，协议的一致性由 IGMP 平台开发团队负责，不在本次系统测试的范围内。

2．测试的质量特性

针对 iBAS R1.0 项目选择的测试类型，主要参考了 ISO/IEC 9126-1:2001 质量模型中定义的质量特性。ISO/IEC 9126-1:2001 质量模型将软件系统的内部质量特性和外部质量特性分为以下 6 类质量特性。

（1）功能性（Functionality）：功能性包括适合性、准确性、互操作性、功能性的依从性（兼容性）和保密安全性。

（2）可靠性（Reliability）：包括成熟性、容错性、易恢复性和可靠性的依从性。

（3）易用性（Usability）：包括易理解性、易学性、易操作性、吸引性和易用性的依从性。

（4）效率（Efficiency）：包括时间特性、资源利用率和效率的依从性。

（5）可维护性（Maintainability）：包括易分析性、易改变性、稳定性、易测试性以及可维护性的依从性。

（6）可移植性（Portability）：包括适应性、易安装性、共存性、易替换性以及可移植性的依从性。

在 ISO/IEC 9126-1:2001 质量模型中，还描述了软件系统的使用质量特性，包括下面 4 个质量特性。

（1）有效性（Usability）：软件产品在指定的使用环境下，使用户能达到与准确性和完备性相关的规定目标的能力。

（2）生产率（Productivity）：软件产品在指定的使用环境下，使用户为达到有效性而消耗适当数量的资源的能力。

（3）安全性（Security）：软件产品在指定的使用环境下，达到对人类、业务、软件、

财产或环境造成损害的可接受的风险级别的能力。

（4）满意度（Satisfaction）：软件产品在指定的使用环境下，使用户满意的能力。

ISO/IEC 9126-1:2001 质量模型有助于定义软件系统的功能性特性和非功能性特性，并选择合适的测试技术、定义软件测试目标和测试出口准则。

☆示例：iBAS R1.0 IGMP 系统测试计划中定义的 IGMP 系统测试需要覆盖的质量特性

iBAS R1.0 IGMP 系统测试计划中定义的 IGMP 系统测试需要覆盖的质量特性如下。

（1）功能性：指软件或者产品在指定条件下使用时，提供满足明确和隐含要求的功能的能力。通过评价特征集和程序的能力、交付的函数的通用性以及整体系统的安全性来评估。可以将功能的准确性、安全性、互操作性、协议一致性和兼容性等都归类到功能性的范畴中。功能性测试在整个测试用例中应该是比例最大的测试类型。

（2）可靠性：重点关注健壮性。健壮性指的是在软件或者产品在异常或者错误的情况下恢复的能力等，例如，系统持续重启测试、测试异常数据的不断输入、异常操作测试、高频率的创建删除操作等。

（3）易用性：指的是软件或者产品在指定条件下使用时，软件产品被理解、学习、使用和吸引用户的能力。易用性是一个非常主观的测试类型，而且在需求中很难定义。因此，这个测试类型需要测试人员了解用户的一般使用习惯和方式。易用性可以包括易理解性、易操作性、易学习性、易安装等。易用性主要是从客户的角度来查看测试对象是否易于理解和使用等。

（4）效率：包括容量、性能和压力测试。容量是指产品或者软件在运行状态下是否能够达到需求要求的最大容量值，例如，IGMP 可以同时支持的组播组数目、每个组播组最大允许的用户数量等。性能指的是软件或者产品在典型的配置运行条件下，它的处理速度、响应时间、资源消耗、吞吐量、效率、丢报率等，此类测试一般需要特殊的测试结果记录，例如，用户加入组播组到收到组播信息的时间间隔。压力测试是指测试系统在极端负载情况下的响应速度和功能表现，例如，IGMP 在最大组播组和最大用户加入的情况下，相关功能是否可以正常运作。

（5）可维护性：重点是稳定性。稳定性指的是在软件或者产品指定条件下使用时，软件产品维持规定性能级别的能力，例如，系统正常配置下，系统长时间运行之后是否表现正常。

（6）可移植性：可移植性指的是软件、产品或者数据等从一种环境转换到另一种环境的能力。可以包含可适应性、可转化性、轻便性和可升级，例如，低版本数据库是否可以方便地移植到高版本、数据库的备份和恢复等。

3．测试时间进度

每个级别测试计划都需要针对该测试级别的测试活动制定测试时间进度。测试活动的测试时间进度并不是一成不变的，而是应该随着项目的深入或者测试活动的持续开展进行不断的更新。

测试时间进度是整个测试估算的重要输出，它的制定依赖于测试工作量估算的结果。

测试估算可以得到测试的规模，例如，设计的测试用例数目、执行的测试用例数目等，而测试工作量的估算是测试估算的重中之重。测试工作量的估算不仅同测试对象的规模有关，同时也和产品的特点、开发和测试过程的成熟度、项目参与者的能力水平、工作产品的质量和测试文档的输出等有关。测试时间进度的制定是根据测试估算的工作量、测试资源、要求的交付时间点等几个方面共同平衡的结果。详细的测试估算知识，请参考 2.5 节。

☆示例：iBAS R1.0 IGMP 系统测试的进度甘特图（部分）

图 2-6 是针对 IGMP 系统测试制定的测试时间进度的部分内容。

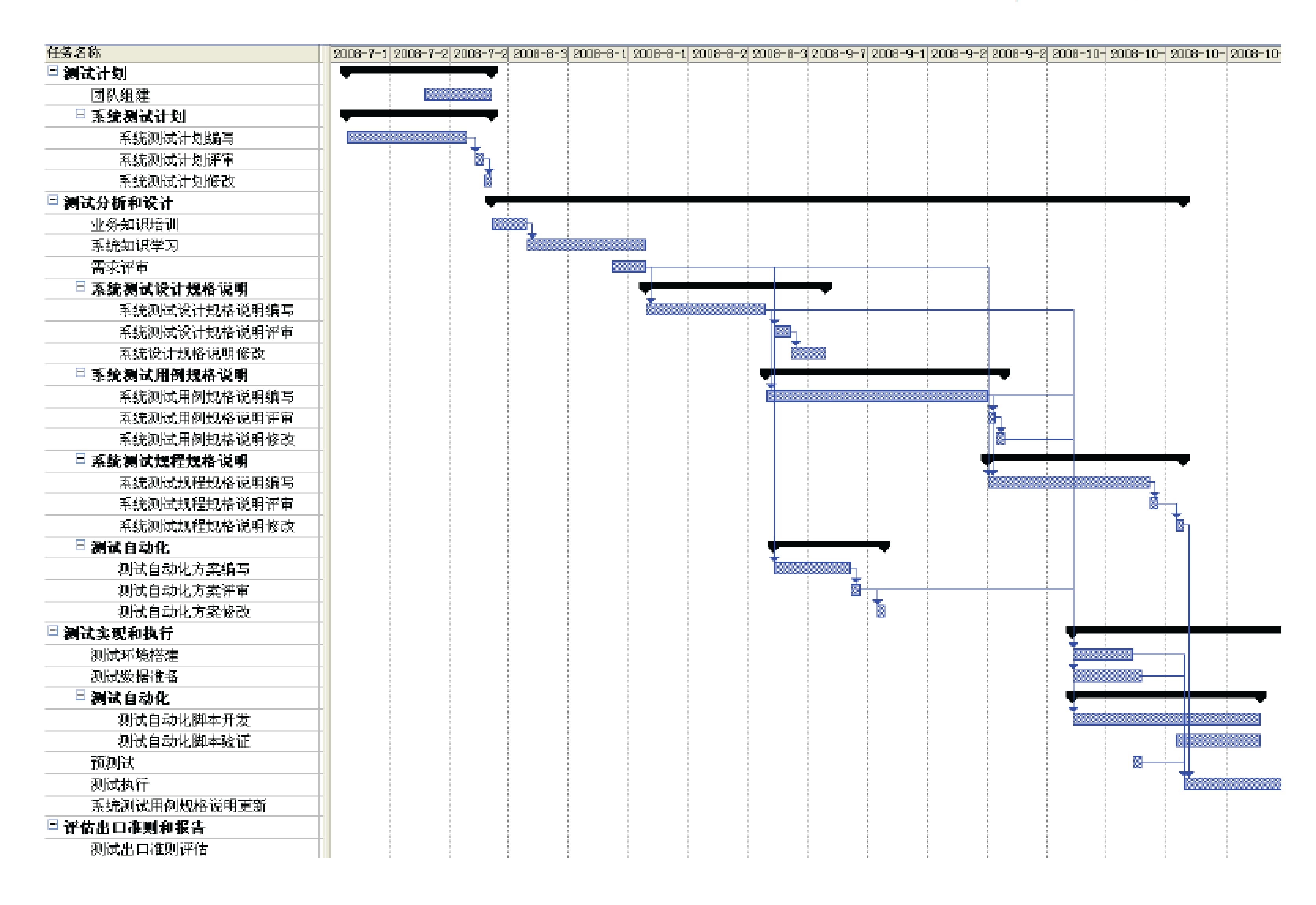


图 2-6　IGMP 系统测试进度甘特图（部分）

4．测试准则定义

为了保证测试的顺利进行，需要在级别测试计划中确定测试的入口准则和出口准则。测试入口准则和出口准则的建立不仅有助于测试的顺利进行，同时也可以帮助测试人员在测试过程中利用这些准则评估和判断测试状态，并根据不同的测试状态采取合适的措施。测试准则并不是测试团队独立制定的，而应该在整个项目范围内进行，由测试、开发和项目管理团队的代表共同参与制定，并在项目组内进行广泛的沟通，从而对测试准则中的要求和条目达成共识。

测试准则根据其目的和作用的不同，可以分为测试入口准则、出口准则、挂起准则和恢复准则，其具体的含义分别如下。

（1）入口准则：允许一个系统进入一个特定的测试阶段时所必须具备的条件，例如，系统测试执行阶段的入口准则可以包括以下内容。

① 系统测试设计规格说明、系统测试用例规格说明、系统测试规程规格说明文档准备就绪，并经过了相应的评审和更新。

② 测试自动化用例和测试脚本经过团队评审和更新。

③ 相关的测试资源和测试环境准备就绪，包括人员、工具、实验室等。

④ 开发人员对提交的版本进行了预测试，并且预测试通过。

⑤ 开发人员提交版本说明，包括该版本中新增加的功能特性、修改的缺陷、没有修改的缺陷、可能存在的问题以及测试重点的建议等。

（2）出口准则：允许一个系统完成一个特定的测试阶段时所必须具备的条件，例如，系统测试执行阶段的出口准则可以包括以下内容。

① 所有在本测试周期所计划的测试用例全部执行完毕。

② 所有的严重缺陷（例如严重程度为 1 和 2 的缺陷）修复完毕，并通过确认测试。

③ 软件测试达到了预先定义的测试覆盖率。

（3）挂起准则：可能会导致测试执行挂起（非正常中止）的状态和事件，例如，测试执行阶段的挂起准则可以包括以下内容。

① 测试环境没有及时到位。

② 测试过程中发现严重问题或者大量问题，以至于继续测试没有任何意义。

（4）恢复准则：可以从挂起点继续或者重新进行测试的状态和事件，例如，测试执行阶段可能的恢复准则包括以下内容。

① 测试环境准备完毕。

② 测试过程中发现的严重问题或者大量问题已经解决，并且符合测试的入口准则。

iBAS R1.0 根据组织和项目的要求，在 IGMP 的系统测试计划中对该级别的测试准则进行了详细的定义。

☆示例：iBAS R1.0 IGMP 系统测试计划中定义的系统测试各种测试准则

iBAS R1.0 IGMP 系统测试计划中定义的系统测试执行入口准则如下。

（1）IGMP 系统测试设计规格说明、IGMP 系统测试用例规格说明、IGMP 系统测试规程规格说明和 IGMP 系统测试自动化用例经过评审并且通过。

（2）IGMP 自动化测试用例在实际的测试环境中验证通过。

（3）IGMP 的集成测试报告已经提交并评审通过。

（4）IGMP 的软件版本在开发人员测试环境中通过预测试。

（5）开发人员提交了 IGMP 的版本说明书，其中阐述了从开发角度建议的测试重点、测试建议、存在的缺陷和问题列表。

（6）涉及 IGMP 测试的测试资源，如测试仪表、测试平台、测试用机、测试人员等全部到位。

iBAS R1.0 IGMP 系统测试计划中定义的系统测试执行出口准则如下。

（1）IGMP 的所有测试用例执行完毕。

（2）IGMP 相关的所有测试用例都处于合适的状态，例如，通过、失败状态，但不能有被阻塞的测试用例。

（3）IGMP 测试用例执行通过率不得低于 95%。

（4）所有状态为失败的测试用例至少有一个缺陷与之进行关联，并且所有的缺陷报告都记录在案。

（5）发现的所有严重程度为 1 的缺陷已经修复（1 为最严重）。

（6）未修复的严重程度为 2 的缺陷不得多于 10 个。

（7）未修复的缺陷总数不得多于 60 个。

iBAS R1.0 IGMP 系统测试计划中定义的系统测试执行挂起准则如下。

（1）无法满足前面定义的测试执行入口准则。

（2）测试团队在执行测试过程中，发现 IGMP 的主要功能没有实现。

（3）测试团队在执行测试过程中，发现 iBAS R1.0 系统经常会出现不可预料的重启、死机等问题。

（4）其他一些影响继续测试的问题。

iBAS R1.0 IGMP 系统测试计划中定义的系统测试执行恢复准则定义如下。

（1）满足了定义的测试执行入口准则。

（2）测试执行挂起准则相关的问题已经顺利解决。

5．测试风险

风险是指会导致负面结果的因素。本书中涉及的风险通常都表达成可能的负面影响。风险根据其性质,又可分为项目风险和产品风险。在测试活动中，项目风险分析的是测试过程的风险，而产品风险分析的是测试对象的风险。

项目风险主要指的是项目管理和控制方面的风险，例如：

（1）组织因素。例如，技能和人员的不足、个人和培训问题。也包括组织高层对产品质量的关注程度、对员工的激励制度等方面的政策因素，例如，与测试人员进行需求和测试结果沟通方面存在的问题；测试和评审中发现的信息未能得到进一步跟踪，以及对测试的态度或预期不合理（如认为在测试过程中发现的缺陷是没有价值的）等。

（2）技术因素。例如，不能正确、完整地定义需求；在现有给定的限制情况下能够满足需求的程度，例如，在经验不足的情况下使用新技术、新工具、不熟悉的编程语言或方法等。

（3）供应商因素。例如，第三方的问题，分包商无法按期、按质地交付产品，合同问题等。

产品风险指的是和测试对象有直接关系的风险，例如：

（1）交付的软件产品运行不稳定。

（2）产品的使用可能会造成相关设备的损坏，甚至人身安全伤害。

（3）低劣的软件质量特性（功能性、可靠性、易用性、效率、可维护性和可移植性）。

（4）软件没有实现既定的功能。

对这些风险的识别和分析结果，通常可以用来决定测试的重点和测试优先级。通过测试，既可以减少风险发生的可能性，也可以降低风险的严重程度。

☆示例：iBAS R1.0 IGMP 系统测试风险列表（部分）

表 2-15 是在测试计划阶段识别的部分 IGMP 系统测试风险。更多关于测试风险的内容，参见 2.3 节。

表 2-15　iBAS R1.0 IGMP 系统测试风险列表（部分）

序号	风 险 列 表	应 对 方 法	责任人	完成时间
1	IGMP 测试仪表没有到位	及时跟踪和催促测试仪表采购过程		
2	IGMP 测试硬件平台缺乏	和开发团队协调，借用部分硬件平台		

续表

序号	风 险 列 表	应 对 方 法	责任人	完成时间
3	IGMP 测试人员无法及时到位	替换测试人员或者相应的备份		
4	IGMP 测试 PC 无法及时到位	测试部门内协调借用		
5	IGMP 估算的测试工作量偏低	加班、增加测试资源		
6	IGMP 用户需求变更频繁	测试经理积极参与变更控制活动		
7	IGMP 项目团队缺乏正规的沟通平台	IGMP 项目定时召开简短沟通会		
8	外包的 IGMP 驱动模块无法及时提交测试	跟踪外包开发的进度		
9	IGMP 需求规格说明质量低下	参与评审工作		
10	IGMP 提交的版本质量低下	定义严格的测试入口准则		

6. 测试文档输出

级别测试计划中需要明确定义在该级别的测试完成后需要交付的测试文档。IEEE Std 829—2008 中定义了测试项目中需要的一些文档，同时测试数据、测试工具和测试自动化脚本等也应该作为测试文档的一部分。测试文档和其他项目开发文档一样，需要进行维护和版本控制。

☆示例：iBAS R1.0 IGMP 系统测试中需要输出的文档

以下是 iBAS R1.0 的 IGMP 系统测试计划中定义的需要输出的文档。

iBAS R1.0 IGMP 系统测试计划。

iBAS R1.0 IGMP 系统测试设计规格说明。

iBAS R1.0 IGMP 系统测试用例规格说明。

iBAS R1.0 IGMP 系统测试规程规格说明。

iBAS R1.0 IGMP 系统测试脚本。

iBAS R1.0 IGMP 系统测试日志。

iBAS R1.0 IGMP 系统测试缺陷报告。

iBAS R1.0 IGMP 系统测试自动化脚本。

iBAS R1.0 IGMP 系统测试报告。

7. 角色和职责

从理论上讲，测试团队的构成和其规模没有多大关系。如项目很小，测试团队可能就一个人，那么这个人就要扮演不同的角色。不同的角色，其职责各不相同。对于测试团队的角色和职责的详细介绍，参见 7.2.1 节。

根据项目或产品的测试级别及可能存在的测试风险，由不同的人充当不同的测试角色，同时保持一定的独立性是合理的。在组件和集成测试的级别，可能由开发人员执行测试，进行验收测试的测试人员一般是业务方面的专家和用户，进行操作性验收测试的一般是将来使用该软件的操作者。

☆示例：iBAS R1.0 IGMP 系统测试中定义的角色和职责

表 2-16 定义了 iBAS R1.0 IGMP 系统测试级别的人员角色和职责。

表 2-16　iBAS R1.0 IGMP 系统测试的角色和职责

角　色	职　责	姓　名
测试经理	测试计划和测试控制专家，具备软件测试、质量管理、项目管理和人员管理等领域的知识和经验	
测试分析师	测试方法和测试规格说明方面的专家，具备软件测试、软件工程以及（形式化）规格说明方法等领域的知识和经验	
测试技术分析师	掌握基于结构的测试技术和静态分析方法，深入了解技术测试的质量特性，熟练掌握测试自动化	
测试自动化人员	测试自动化专家，具备测试基础知识、编程经验，以及丰富的测试工具和脚本语言知识。利用项目中提供的测试工具，按需要进行测试的自动化	
测试环境管理员	安装和操作测试环境方面的专家，具备系统管理员知识。建立和支持测试环境，需要经常与系统管理员和网络管理员进行协调	
测试执行人员	执行测试和事件报告方面的专家，具备 IT 基础知识、测试基础知识，能应用测试工具，熟悉被测试对象	

2.4.5　项目风险管理

软件项目计划的一个重要组成部分是处理项目风险。项目风险的识别可以使用 2.3 节中所描述的类似过程。项目风险的主要责任人是项目经理，因此测试团队识别项目风险之后，应提交给项目经理，并由他制定应对措施。尽管有些项目风险的缓解并不是测试团队能处理的，但是有些项目风险可以，而且也应该由测试经理进行应对和缓解，例如：

（1）测试环境和工具是否及时就绪的风险；

（2）测试人员的可用性和能力方面的风险；

（3）缺乏测试工作的标准、规则和技术的风险。

管理项目风险的方法包括尽早准备测试件、尽早验证测试环境、预测试软件产品的早期版本、使用更严格的测试入口准则、加强需求的可测性、参与项目早期工作产品的评审、参与对变更的管理，以及监控项目的进展和质量等。

在识别和分析了项目风险后，项目风险的管理可以采用以下 4 种方案。

（1）缓解风险，通过预防性措施降低风险的可能性或严重程度，或者两者同时降低；

（2）制定应急计划，降低风险成为事实后的影响程度；

（3）将风险转移给第三方来处理；

（4）忽略或接受风险。

软件开发生命周期中具体采用哪种项目风险管理手段，取决于该方案为各方利益干系人带来的收益、机会、花费成本，以及和它们相关的潜在的任何额外风险。如果为某个项目风险确定了应急计划，最佳实践是同时确定应急计划的触发条件（以此确定触发应急计划的时机和方式）和负责应急计划的责任人。

2.4.6　其他的测试工作产品

除了前面提到的测试方针、测试策略、主测试计划和级别测试计划之外，测试过程中

还会生成其他的测试工作产品，例如，测试用例规格说明、测试设计规格说明、测试规程规格说明、测试日志和缺陷报告等。通常测试分析师和技术测试分析师会负责编写这些测试工作产品。在测试分析师高级大纲中，有专门章节讨论生成和编制这些测试工作产品需要考虑的因素。测试经理需要确保测试工作产品在下列测试活动中保持一致和质量。

（1）构建和监督用来监视这些测试工作产品质量的度量，例如，被拒绝的缺陷报告的百分比；

（2）与测试分析师、技术测试分析师一起选择合适的测试工作产品模板，并根据要求进行相应的裁剪；

（3）与测试分析师、技术测试分析师一起构建合适的测试工作产品的标准，例如，测试用例、测试日志和测试报告所需的详细程度；

（4）选择合适的项目利益干系人，并采用合适的技术参与测试工作产品的评审。

测试工作产品的模板的来源有很多，本大纲主要参考的是 IEEE Std 829—2008 版本。该标准的模板适合任何行业，因此模板中包含很多非常详细的内容，其中一些内容并不一定适合特定的行业、组织或者产品。因此，测试经理需要根据特定产品和组织等特点对 IEEE Std 829—2008 进行裁剪。统一的文档模板有助于减少培训需求，并且帮助组织内部统一过程。

测试过程中具体选择什么样类型的测试文档、测试文档的范围和特性，主要取决于选定的软件开发生命周期、可用的标准和法规、开发的特定软件系统的产品质量和项目风险。

2.5 测试估算

在测试计划和控制阶段，测试经理的一个重要的测试活动是测试估算，通常需要测试分析师和技术测试分析师的支持。测试估算得到的结果是制定测试计划的基础，同时也是测试资源安排、测试过程监控的主要输入。测试估算的正确与否，直接影响了测试活动的开展。尽管进行精确的测试估算很难，但是测试经理应该尽量在已有的条件下，利用合适的方法和技术降低测试估算的误差，以满足不同测试阶段的要求。

测试估算过程中经常会涉及不同的概念，例如，测试规模、测试工作量和测试进度等，它们的含义是不同的。测试规模通常指的是测试用例的数目，而一般不用开发工作量估算中的代码行或者功能点表示。测试工作量通常是基于测试规模展开的，但并不是所有的估算技术都是基于测试规模的，例如，基于百分比的估算方法。测试工作量的单位可以采用人·年、人·月、人·周或者人·天来表示（在本节中，测试工作量采用的单位是人·周）。

测试估算是系统化和持续的过程，应该在项目启动的早期就开始。在对软件产品有所了解的情况下就可以开始进行粗略的估算，随着对系统了解的深入和相关文档的不断完善，在后续阶段可以进行更加精确的估算，从而对前面的粗略的估算进行相应的更新和修正。如图 2-7 所示，随着测试活动的不断开展，测试估算越来越接近最终的实际结果。

高精度的测试估算是很困难的，也不是必需的。针对测试对象进行合理的估算可以帮助测试经理制定合理的计划，使测试活动尽量满足测试计划，并在可控的范围内对测试计划进行更新和调整。实际上，在对测试对象进行估算的时候，通常很难判断估算的精确性。因为估算的准确与否，只能通过测试对象的实际数据与估算得到的数据进行比较之后，才

能进行判断。

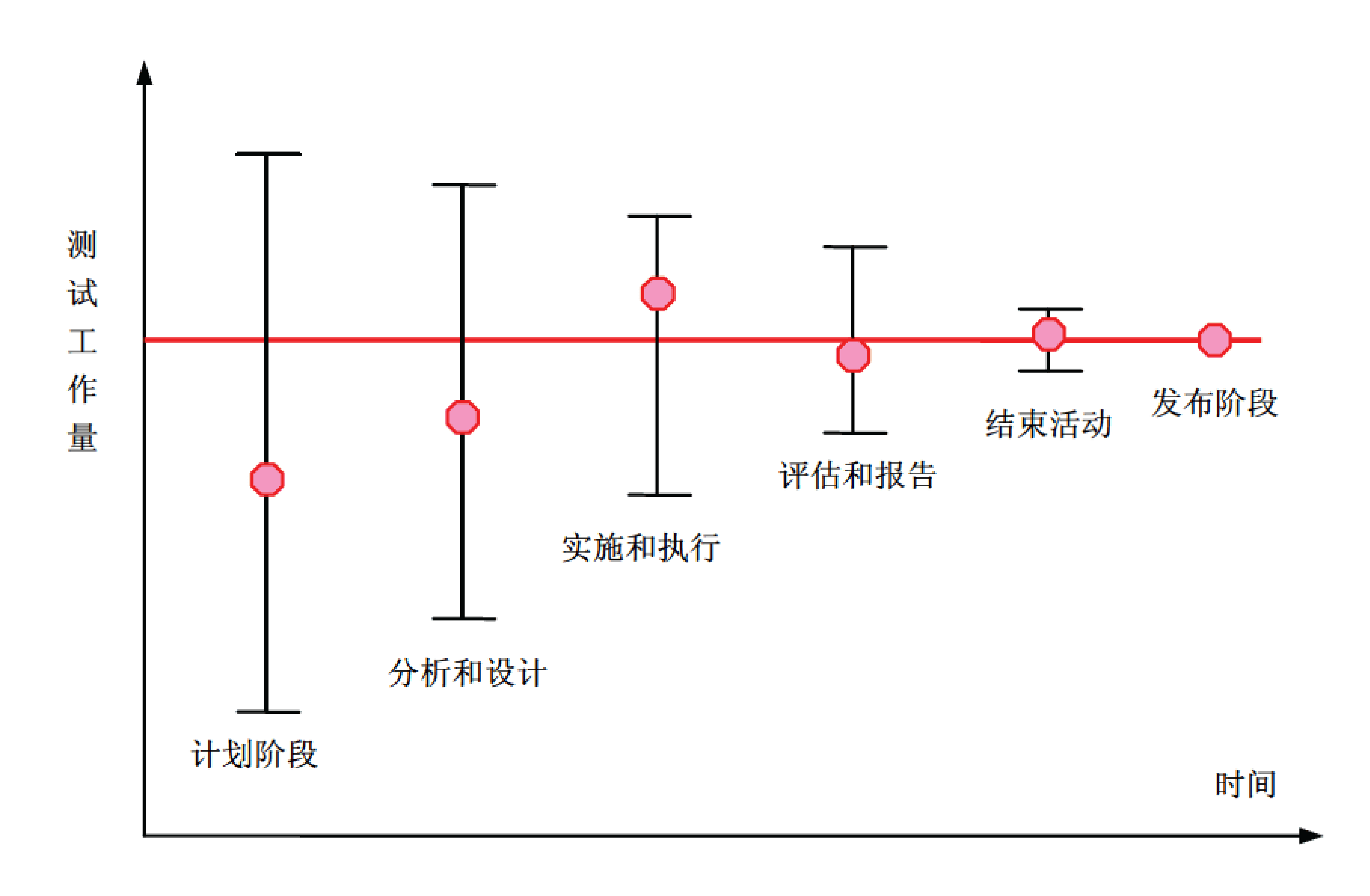


图 2-7　估算阶段和估算精度

既然测试估算非常困难，那么什么是一个好的估算呢？测试估算作为测试管理活动之一，是为了获得特定业务或项目中各种活动的近似成本预算、工作量估算和完成日期。因此，好的估算应该具备以下特点。

（1）估算结果代表有经验的同行的集体智慧，并能得到项目利益干系人的支持；

（2）估算结果可以对测试对象和测试过程提供全局的视野，例如，明确而详细的资金、资源、任务和项目利益干系人列表等；

（3）估算结果可以保证测试团队对时间、成本、范围等的承诺处在一个合理的范围；

（4）估算结果是针对每个估算的活动提供最可能的成本、工作量和持续时间；

（5）估算结果有助于整个测试过程的有效监控和管理；

（6）估算结果有助于更好地实现测试目标，在测试估算、测试承诺和测试目标之间达到一个较好的平衡。

测试估算可以自底向上进行，也可以自顶向下进行。在测试估算中，通常可以使用下列技术。

（1）直觉和猜测；

（2）过去的经验；

（3）工作分解结构（Work Breakdown Structure，WBS）；

（4）团队评估会议（例如：Wide Band Delphi 法）；

（5）三点估算方法；

（6）测试点分析（Test Point Analysis，TPA）；

（7）公司标准和规范；

（8）测试占整个软件项目工作量的百分比或人员编制比例（例如：测试人员与开发人员的比例）；

（9）组织的历史数据和度量标准，包括基于度量标准的模型，该模型用于估算缺陷数、测试周期数、测试用例数、每个测试的平均工作量以及回归测试周期的数目等；

（10）行业平均值和预估模型，例如，测试点、功能点、代码行数、估计的开发工作量或其他项目参数等。

在多数情况下，测试团队得出测试估算结果后，需要将结果提交给管理人员，并说明测试的价值（详细内容可以参考 2.7 节）。理想情况下，最终的测试估算结果平衡了组织和项目在质量、时间、预算以及特性方面的要求。为了有效地开展测试估算工作，得到合理的测试规模和测试工作量，以帮助确定和安排测试资源、测试持续时间、测试成本，测试经理首先需要了解影响测试估算的主要因素。

2.5.1 测试估算的影响因素

测试估算应针对测试过程中的所有阶段和测试活动（例如：测试计划和控制、测试分析和设计、测试实施和执行、评估出口准则和报告以及测试结束活动）。由于测试执行通常在项目的关键路径上，测试估算的成本和工作量，特别是测试执行持续时间是管理人员特别关注的。然而，当软件产品的整体质量很低或还未知的时候，对测试的估算会比较困难，并且估算得到的结果也不可靠。另外，估算过程中所做的假设应该作为测试估算的一部分并文档化。

测试估算应考虑影响测试活动的成本、工作量和持续时间等所有因素，这些因素包括（但不仅限于此）：

（1）软件产品所要求的质量级别。

（2）被测试系统的规模。

（3）以前测试项目的历史数据（也可包括基准数据）。

（4）过程因素：包括测试文档的编写、维护测试、过程成熟度，以及软件产品对估算准确度的要求。

（5）环境因素：包括测试自动化和工具、测试环境、测试数据、开发环境、项目文档（例如：需求规格说明、设计规格说明等），以及可重用的测试工作产品等的要求。

（6）人员因素：包括技术负责人和管理人员的承诺和期望、项目团队的技能、经验和态度、项目团队的稳定性、项目团队的关系、测试和调试环境的支持、是否有合格的承包商和顾问以及相关领域的知识。

其他因素也会影响测试估算，包括过程的复杂度、采用的技术、组织、测试项目干系人、测试子团队（特别是当子团队处于不同的地理位置）；项目启动、培训、定向需求；新工具的熟悉和开发、定制的硬件、测试件的数目；非常详细的测试规格说明的要求（特别是使用一个不熟悉的文档标准时）；难以确定的组件交付时间，以及敏感的测试数据（例如：对时间敏感的数据）。通过对上面的各种因素分析，可以将影响测试估算的因素归纳为以下 6 个方面。

1．产品的特点

软件产品开发的目的是为用户提供某种服务或者某种产品，因此，开发的软件系统应用领域的复杂度、可靠性和安全性等方面的需求是测试估算需要考虑的重要因素，例如，

安全关键系统和游戏软件对测试工作的要求是完全不一样的（针对安全关键系统需要进行更加严格和全面的测试），其关注的测试重点也不一样，导致测试工作量的估算结果也不一样。因此，针对不同产品的特点，在估算测试工作量的时候需要区别对待。

2．测试的规模

软件产品或者基于软件的系统的测试规模是进行测试工作量估算的基础和重要输入。在进行测试估算时，需要详细考虑被测软件产品的规模。测试规模至少需要从下面几个方面进行考虑。

（1）需要新设计的测试用例的数目；

（2）需要执行的新的测试用例的数目；

（3）需要进行的回归测试用例的数目；

（4）测试对象中可能存在的缺陷数目，以及因这些缺陷而引起的确认测试和回归测试的工作量。

3．工作产品质量

测试估算是基于软件开发过程中得到的工作产品而展开的，因此，工作产品的质量直接会影响测试的估算。下面是在测试估算过程中针对工作产品质量需要考虑的几个方面。

（1）开发文档的质量：是指在开发过程中输出的相关文档的质量，如系统需求规格说明、概要设计规格说明、详细设计规格说明等。由于这些开发文档是测试活动（例如，测试设计和测试执行）的基础，它们的质量情况会直接影响测试估算的正确性和有效性。如果系统需求规格说明中只定义了功能性需求，而遗漏了非功能性需求，则会导致在测试估算中遗漏这方面的工作，从而影响测试估算的正确性。

（2）测试文档的质量：一方面开发文档的质量会影响测试文档的质量，另一方面，测试文档的质量也会影响后续的测试执行的质量。测试文档包括测试计划、测试设计规格说明、测试用例规格说明、测试规程规格说明等。测试人员的技能水平和在项目相关领域的背景知识的不同，都会影响测试文档的质量，例如，测试计划中没有确定测试方法、测试技术和测试范围等，直接会影响测试估算的正确性和有效性。

（3）软件代码的质量：开发文档的质量和开发人员的技能、知识水平会影响软件代码的质量。而软件代码是测试过程中重要的测试对象之一，它的质量高低直接影响了测试的执行。如果代码质量很差，那么测试执行可能并不是原来计划中的一次，而会是多次。同时，差的代码质量也意味着其中存在较多的缺陷数目，从而会增加缺陷相关的确认测试和回归测试的工作量。

（4）测试对象中可能存在的缺陷数目以及后续的确认测试和回归测试的工作量，这个在测试估算过程中经常容易被忽视。测试执行过程中发现的缺陷在修改完成以后，需要测试人员进行确认测试和相关的回归测试；假如在测试计划中没有考虑这些测试活动，常常会降低测试估算的精度，从而导致测试后期的测试任务非常繁重。

4．项目参与者

项目参与者包括项目系统人员、开发人员、测试人员以及管理人员等，是测试估算的重要输入。不同项目参与者的能力是不同的。在项目团队比较稳定的情况下，测试估算可以从该团队的历史信息中获得一些经验数据，从这些数据中可以得到测试组织的生产力水平和能力，例如，测试用例执行的速率（例如，执行的测试用例数目/星期）、测试用例的

有效性（发现的缺陷数目/测试用例数目）等。

5．过程成熟度

组织的过程成熟度是影响测试估算的又一个重要因素，例如，开发过程的成熟度和测试过程的成熟度。软件产品的质量很大程度上取决于整个组织的过程成熟度，而不仅仅是某个人或者某部分人的能力。因此，组织的过程成熟度直接会影响测试估算，具体表现在：

（1）开发过程成熟度，直接决定了开发得到的工作产品的质量，例如，软件的需求规格说明、设计规格说明、代码等工作产品的质量。软件开发得到的工作产品都可以是测试对象，同时它们也是测试的基础。这些工作产品的质量会直接影响测试估算。

（2）测试过程成熟度，决定了主要的测试阶段和活动。不同的测试过程成熟度，需要执行的测试活动是不一样的，例如，某些组织的测试执行阶段，包括正式测试执行之前的预测试；有的组织的测试执行阶段，可能强调回归测试。同时，测试过程成熟度高的组织，可以避免一些重复的工作。因此，测试过程的定义和成熟度也会影响测试的估算。

（3）组织级别的度量数据，反映了该组织的能力，例如，组织的测试用例执行的速率（测试用例数目/星期）、测试用例的有效性（缺陷数目/测试用例）等，这些度量数据是进行测试估算的重要输入。

6．测试的交付物

测试过程中需要输出一些测试相关的文档，如测试计划、测试设计规格说明、测试用例规格说明、缺陷报告和测试报告等，这些文档的完成都需要一定的工作量。缺陷报告和测试报告的工作量在测试估算中常常容易被忽视。在测试执行过程中，缺陷报告是测试团队重要的输出。在测试执行中发现缺陷以后，需要测试人员去重现和确认发现的缺陷，并以书面形式提交给开发人员，对缺陷的状态进行跟踪、管理和验证；测试总结报告或者测试报告指的是对软件系统进行测试产生的行为及结果的描述文件。测试总结报告以文档的形式描述了被测软件的测试情况和测试结果，并对相关的结果和数据进行分析，向项目管理层提供信息和建议。

上面分别从产品的特点、测试的规模、工作产品质量、项目参与者、过程成熟度和测试的交付物 6 个方面讨论了影响测试估算的因素，测试估算应该针对这些因素进行综合考虑，才可能得到比较合理的测试估算结果。下面是测试过程中经常采用的几种测试估算方法。

（1）基于百分比的方法；

（2）基于专家团队的方法；

（3）基于类似项目的方法；

（4）基于工作分解结构的方法。

2.5.2 基于百分比的测试估算

最简单的测试工作量估算方法是根据组织中测试占整个软件开发生命周期活动工作量的比例来进行估算的，例如，测试的工作量占整个项目工作量的 50%，如表 2-17 所示。这种估算方法虽然简单，但是应该比没有进行任何估算的情况好得多，尤其是当测试团队所面临的项目是团队成员没有任何相关经验的情况下，这是比较好的一种方式。

表 2-17 基于百分比的测试估算

编　号	开 发 活 动	百分比	工作量/PW
1	用户需求	13.00%	16.25
2	概要设计	12.00%	15
3	详细设计	15.00%	18.75
4	编码和实现	10.00%	12.5
5	测试活动	40.00%	50
6	项目收尾	10.00%	12.5
总共		100.00%	125

对测试团队而言，基于百分比的测试工作量估算方法在估算方面需要花费的精力有限，因为测试工作量是直接通过软件项目整体的工作量的百分比而得到的。基于百分比的估算方法需要采用很多以前项目的经验数据。同时与组织内采用的软件开发生命周期和测试过程紧密相关，例如，测试过程是否贯穿于整个开发生命周期、测试过程是否采用静态测试等。

2.5.3　基于专家团队的测试估算

在项目的计划阶段，例如，用户需求阶段，需要进行项目工作量估算，而测试工作量作为整个项目工作量的重要组成部分，必定也需要在早期提供一个粗略的工作量数据。通常情况下，早期的测试工作量数据是由测试经理进行估算的。由于基于单个专家（例如测试经理）进行的测试工作量估算容易存在这样的问题：测试的工作量估算值要么完全高估，要么完全低估（运气好的时候当然也可能得到比较正确的估算），无法体现基于团队估算的概率统计分布方面的优势。

基于专家团队的估算方法是多个估算专家（专家团队）共同进行估算的一种方法。尽管团队中的每个专家得到的估算结果都可能出现估算偏高和偏低的情况，但是由于存在概率统计分布方面的优势，得到的最后估算结果总体上可以在一定程度上进行平衡，从而相对提高工作量估算的精度。基于专家团队的估算方法能够保证估算结果体现整个团队的估算经验和能力，从而避免单个专家的估算结果偏离对整个估算结果造成的影响过大。

基于专家团队的估算方法要求测试估算团队的每个成员对测试对象的工作量得到一个自己认为合适的估算值，然后基于团队的方式对每个估算专家得到的结果进行分析和评估，并且最终得到大家都同意的一个收敛的测试工作量估算结果。这种基于团队的测试工作量估算方法，有时候也称为宽带德尔菲法（Wide Band Delphi 法）。Wide Band Delphi 估算方法的具体步骤如下。

（1）估算召集人（例如测试经理）给每个估算专家阐述估算的对象，以及可能的一些基本假设条件和前提条件。

（2）每个估算专家独立地准备测试工作量估算值。

（3）每个估算专家以匿名的方式将自己的估算值提交给估算召集人。

（4）估算召集人对每个估算专家的估算值进行收集和整理，并以图表的形式表示，每个估算专家可以清楚地看出自己的估算结果所处的位置和范围。

（5）估算召集人组织召开针对工作量估算值的讨论会，讨论各自估算值存在的偏差，以及各自的理由。

（6）通过匿名投票的方式，确定估算专家是否同意当前的平均估算值，或者当前的估算范围。

（7）假如其中有专家不同意，就再次重复上述的步骤，直到最后大家能够对测试工作量估算值或者估算范围达成一致。

图 2-8 是专家团队对某个测试活动的工作量（单位是人 • 月）通过三轮的估算和讨论，最终达成一致的测试工作量估算值（估算范围）的一个例子。

Wide Band Dephi 测试工作量估算方法的运用需要避免其中的估算专家对整个估算结果有太大的影响能力，例如，项目经理，他可能有意识地将估算结果和项目的目标结合起来。基于专家团队的估算方法和基于单个专家的测试工作量估算方法相比，尽管可以得到更加合理的估算值，但是专家团队的方法需要花费更多的时间。因此，需要的成本更大。

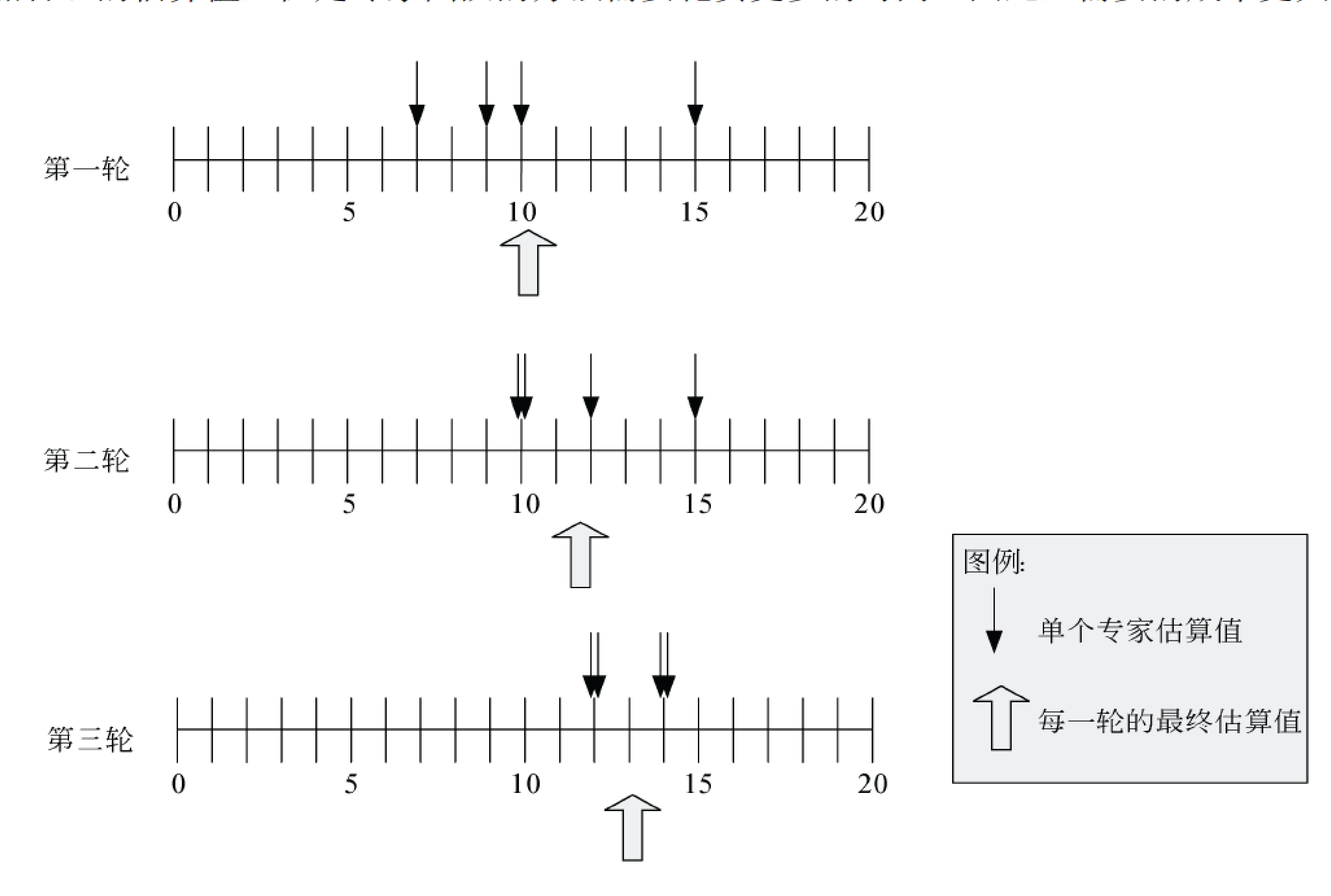


图 2-8 Wide Band Delphi 工作量估算方法[①]

2.5.4 基于类似项目的测试估算

基于类似项目的测试工作量估算方法是通过将新项目和以前类似的项目进行比较，从而得到相对精确的测试估算值。在运用基于类似项目的测试工作量估算方法的时候，需考虑下面的一些因素。

（1）参考以前项目的历史数据和经验必须适合当前项目的任务，例如，必须考虑公司

① Steve McConnell．Software Estimation: Demystifying the Black Art．Microsoft Press,2006.

采用的不同软件开发技术，采用 C 语言进行嵌入式软件开发的工作量估算数据，不一定会适用于采用 Java 开发的 Web 产品。

（2）需要检查以前类似项目的历史数据的真实性，特别是测试数据的真实性。

（3）假如条件允许，尽量使用多个不同的历史项目数据。如果项目之间的一些历史数据存在矛盾，可以通过取平均值或者标准偏差方式进行相应的调整。

类似项目的经验数据有助于当前项目的测试估算。为了整个组织能够更加有效地利用这种方法，测试经理需要在测试过程中不断收集各个测试活动实际花费的工作量。在整个测试活动结束以后，通过比较总的测试活动工作量和原来估算的工作量，可以得到一个更加准确并且更加符合这种项目类型的测试工作量比例，这样就可以为将来的项目提供更加合理和准确的基于经验的测试工作量比例，从而更加合理地安排测试资源和测试活动。

2.5.5　基于工作分解结构的测试估算

基于工作任务的分解（工作分解结构 WBS）的估算方法，测试经理首先需要将测试项目分解成不同的测试活动。针对每个不同的测试活动或者测试任务估算它们各自的工作量，所有测试活动的工作量之和就是整个项目的总的测试工作量估算。

尽管软件规模大小对很多项目活动的工作量估算也起到重要的作用，但是基于工作分解结构的方法并不明确需要软件的规模大小，相反，它需要的是测试对象相关的工作任务列表。一般而言，确定工作任务列表相对比较容易。基于工作分解结构的估算方法的一个风险是可能在工作任务列表中遗漏了一些工作任务。下面列出了测试过程中一些常见的测试活动。

（1）测试计划：制定测试计划的工作量和时间要求，以及进行整个测试活动工作量估算需要花费的工作量。

（2）测试准备：测试对象配置管理、测试对象的安装、测试文档和模板的配置管理、测试数据库的准备以及测试环境的准备等。

（3）测试人员培训：包括测试技术、测试过程、产品技术和相关的测试工具等方面的培训，以及相关的测试技术和测试工具学习等。

（4）所有测试级别的测试规格说明输出：包括测试用例的分析和设计、测试用例的实现等。

（5）测试用例实现自动化。

（6）测试执行过程中测试文档的输出：包括测试缺陷报告和测试总结报告等。

（7）测试执行、确认测试和回归测试：测试过程中需要执行的测试，以及估算可能发现的缺陷数目和修正的缺陷数目，针对这些缺陷需要进行的确认测试和回归测试的工作量。

（8）测试度量相关的数据收集和分析。

（9）项目成员之间的沟通：包括 E-mail 沟通、项目状态会议、缺陷报告会议、变更控制会议和部门会议等。

测试经理必须明确地识别所有的测试活动，并且进行单独的估算。测试活动需要尽量细化，避免出现单个测试活动占用过多工作量的情况。假如分解的测试活动粒度太大，可能会导致估算偏差较大，同时也不利于测试过程的监控。

2.6 定义和使用测试度量

测试过程中，经常会碰到一些需要通过测度来解决的问题，例如，项目经理希望知道测试是否能够按时完成、版本质量如何、是否能够按时发布；测试经理在制定测试计划时如何保证计划的准确性等。这些问题都需要通过测试度量才能够比较准确地给出答案，避免由于“拍脑袋”而导致决策错误①。

“只有测试度量才能帮助预测或者控制你的工作”②这句话很精辟地说明了测试度量的重要意义。使用度量有助于了解项目当前的状态，可以更好地指导后续的测试活动，可以让测试人员在汇报结果时保持一致，并对测试进度和结果进行连贯地跟踪。测试经理通常会在各种项目会议上展示各种度量数据，会议的参与人包括开发人员、执行管理人员和各级别的用户等。判断测试项目成功与否，也需要用到这些度量数据。同时，测试经理需要确定跟踪什么内容、汇报的频率和呈现这些信息的方式等。

测试度量本身并不能触发过程和质量的改进，测试度量需要有明确的目的，测试度量的结果需要转化为合适的应对措施和活动。测试经理在应用测试度量过程中，必须考虑以下内容。

（1）度量的定义：对这些定义的度量的解释必须得到项目利益干系人的认可，避免将来使用这些度量时产生争议。度量可以根据过程、任务的目标、面向系统组件、个人或团队进行定义。在实际测试过程中经常存在定义了过多的度量，而忽略了那些最重要度量的情况。

（2）度量的追踪：尽可能把度量报告和集成的工作自动化，以减少生成原始度量数据所用的时间。某些特定的度量数据可能会随着时间的变化带来其他信息，这些信息可能没有包含在度量的定义中。测试经理应做好准备，仔细分析这些度量数据和期待可能出现哪些偏差，以及造成偏差的原因。

（3）度量的汇报：度量的目的是帮助管理人员迅速理解软件项目当前的状态信息和数据。度量汇报的结果应呈现对软件产品某段时间度量的“快照”或度量随时间推移的变化，这样才能进行趋势分析。

（4）度量的有效性：测试经理还必须验证汇报的度量信息。为某个度量收集的数据可能无法反映项目的真实情况或可能传达了过于乐观或过于悲观的趋势。在呈现数据前，测试经理必须就数据的准确性和可能传达的信息两方面对数据进行评审，以避免没有关注那些最相关的度量的情况。

根据目的不同，测试度量可以划分成不同的类型，主要包括：

（1）项目度量，根据既定的项目出口准则度量项目进展，例如，测试用例执行率、通过率和失败率等；

---

① 目前国内对 Measurement 和 Metrics 的翻译比较混乱，本文中对这两个词汇的翻译来自 ISTQB 的中文术语表，详细的术语的定义可以参考 CSTQB 的官方网站 www.cstqb.cn。读者在阅读过程中应该将重点放在对内容的理解上，避免在术语翻译上过多的纠结。

② N Fenton，S Pfleeger．Software Metrics（2nd ed.）．PWS Publishing，1997.

（2）产品度量，度量产品的某些属性，例如，已完成的测试范围和深度、缺陷密度等；

（3）过程度量，度量测试或开发过程的能力，例如，通过测试检测的缺陷百分比；

（4）人员度量，度量个人或团队的能力，例如，给定时间内测试用例的实施情况。

测试过程中给定的任何度量，都可以归类到上述几种不同类型，例如，体现每日缺陷发现率的趋势图可以与以下内容相关：测试执行的出口准则（例如，连续一周内都没有发现新的缺陷），体现的是项目度量；产品质量（例如，测试无法再发现软件产品中的缺陷），体现的是产品度量；测试过程能力（例如，测试执行前发现了大量的缺陷），体现的是过程度量。

测试过程中的人员度量是特别敏感的度量，在测试实践中应该特别注意。测试经理有时会将过程度量误认为是人员度量，导致测试人员为了让该度量对他们更有利而采取一些行动，从而产生严重的后果。本书第 7 章和专业测试经理大纲（专家级）中讨论了对测试人员的适当激励和评估。

测试过程中不断收集和分析测试数据是测试人员，特别是测试经理在测试管理过程中的一项非常重要的工作。测试数据的收集和分析也是测试经理实现测试过程监控的重要输入。测试经理需要根据测试数据不断地对测试资源和测试人员等进行协调和重新分配，或者根据测试状态更新测试计划；同时测试过程中收集到的测试数据也可以为将来项目开发和测试的改进提供各种不同的建议和经验教训。

高级测试经理大纲主要关注的是应用度量评估测试工作的进展，如项目度量。某些项目度量也会与产品度量和过程度量相关。监控测试进展的度量主要包括：

（1）产品风险（质量风险）；

（2）缺陷；

（3）测试；

（4）覆盖率；

（5）信心。

产品风险、缺陷、测试和覆盖率度量数据，假如与测试计划中定义的出口准则相关，它们可以作为判断测试工作是否完成的客观标准。信心的度量数据可以通过调查或者使用覆盖率作为替代度量，通常是以主观的方式进行汇报。

2.6.1　产品风险

产品风险是测试计划的重要组成部分，同时也是整个测试过程中需要不断进行跟踪和更新的对象。作为测试经理，应该始终将产品风险作为整个测试管理的一个关注点。与产品风险相关的度量主要包括：

（1）测试为“通过”状态所覆盖的产品风险百分比；

（2）测试为“失败”状态所覆盖的产品风险百分比；

（3）测试为“未完成”状态所覆盖的产品风险百分比；

（4）按风险级别排序得到的已覆盖的产品风险百分比；

（5）前期产品风险分析之后再识别的产品风险百分比。

案例分析：风险度量在测试进度监控中的应用

在 iBAS R1.0 项目中，针对 IGMP 功能，测试经理召集了不同角色的人员对测试过程中可能存在的风险进行了讨论。风险识别过程中采用了不同的技术和方法，例如，风险模板、以前项目的经验教训、头脑风暴法等。表 2-18 是针对 IGMP 功能的测试风险列表，包括风险发生的可能性和严重程度，以及两者乘积得到的测试风险级别。

在 iBAS R1.0 项目中，假定测试风险级别为 1 及其以下值是可以接受的。假如通过测试风险应对，使其测试风险级别在 1 或者 1 以下，那么可以认为该测试风险已经成功进行了应对，其状态可以更改为关闭状态（或者可接受状态）。测试风险列表主要包括三个部分：识别的测试风险、初始的测试风险级别和当前的测试风险级别。

表 2-18　IGMP 风险列表

ID	IGMP 风险列表	初始值			当前值		
	描述	风险级别	可能性	严重程度	风险级别	可能性	严重程度
1	资源						
1.1	IGMP 测试仪表没有到位	4.5	50%	9	0	0	9
1.2	IGMP 测试硬件平台缺乏	2.7	30%	9	0	0	9
1.3	参与 IGMP 测试的人员目前还在其他项目测试任务中，无法及时介入 IGMP 测试工作	2.1	30%	7	1.4	20	7
1.4	IGMP 测试人员需要参与其他更高级别的测试任务，例如：产品的入网测试	1.4	20%	7	0	0	2
1.5	控制 IGMP 测试的服务器/PC 缺乏	0.5	10%	5	0	0	5
2	技术						
2.1	IGMP 测试范围的变更，由于需求文档中缺少非功能性需求的描述	3	60%	5	0.6	20%	3
2.2	测试团队缺乏 IGMP 测试经验	1.2	30%	4	0.3	10%	3
2.3	IGMP 测试工作量的偏低估算，有些测试活动没有估算在内	4.2	70%	6	1.2	30%	4
2.4	用户的 IGMP 需求经常发生变更	1.4	35%	4	0.4	20%	2
3	质量						
3.1	系统需求规格说明和设计规格说明评审过程中发现大量的缺陷	4.2	70%	6	0.3	10%	3
3.2	测试计划没有经过详细的评审	1.5	30%	5	0.6	15%	4
3.3	测试设计规格说明没有经过详细的评审	1	20%	5	0.4	10%	4
4	沟通						
4.1	需求人员、开发人员和测试人员之间缺乏正规的沟通平台	1	50%	2	0	0	2
4.2	测试人员测试中发现的缺陷，开发人员无法及时到现场进行确认	1	20%	5	0	0	2
5	第三方/外包						
5.1	IGMP 相关的驱动部分以外包形式进行开发，是否能及时提交？以及提交的质量	1.5	30%	5	0	0	2

续表

ID	IGMP 风险列表	初始值			当前值		
	描述	风险级别	可能性	严重程度	风险级别	可能性	严重程度
6	法律						
6.1	某测试工具软件，测试团队只有两个 License。在测试过程中要保证只有两台机器安装了该工具软件，避免不必要的版权纠纷	1.4	70%	2	0	0	2

针对 IGMP 测试风险，主要用测试风险管理指标（TRMI）、测试风险减轻有效性（TRME）和测试风险趋势（TRT）三个指标进行风险的分析，发现其中可能存在的问题，并根据分析结果制定应对措施。测试风险管理指标（TRMI）和测试风险减轻有效性（TRME）都是在项目结束之后才可以通过计算得到的，以此评估测试风险管理是否满足组织的要求。项目结束后，由 IGMP 风险列表以及在每个测试阶段的里程碑点收集的数据得到如下信息。

（1）测试计划阶段里程碑点 M1 之前发现的测试风险个数 R_1：12 个。

（2）测试分析和设计阶段里程碑点 M2 之前新发现的测试风险个数 R_2：2 个。

（3）测试实施和执行阶段里程碑点 M3 之前新发现的测试风险个数 R_3：2 个。

（4）测试评估和报告阶段里程碑点 M4 之前新发现的测试风险个数 R_4：0 个。

（5）关闭的测试风险个数：14。

（6）没有关闭的测试风险个数：2。

（7）原始的累计测试风险值。

（8）应对之后的累计测试风险值。

根据前面收集的这些数据，可以计算得到测试风险管理指标（TRMI）为

$$\text{TRMI}=\frac{R_1}{R_1+R_2+R_3+R_4}\times 100\%=\frac{12}{12+2+2+0}\times 100\%=75\%$$

测试风险减轻有效性 TRME 为：

$$\text{TRME}=\frac{\text{总共关闭的测试风险数目}}{\text{总共关闭的测试风险数目}+\text{没有关闭的测试风险数目}}\times 100\%$$
$$=\frac{14}{16}\times 100\%=87.5\%$$

总的测试风险趋势（TRT）如图 2-9 所示。

有了前面的三个测试风险相关度量：测试风险管理指标（TRMI）、测试风险减轻有效性 TRME 和测试风险趋势（TRT），就可以对测试风险的现状和趋势进行分析，并根据测试风险在测试过程中的状态和测试风险值，采取合适的应对计划。

IGMP 测试过程中，分别在测试分析和设计、测试实施和执行阶段中识别了 4 个比较重要的新的测试风险，并对测试相关的活动产生了影响，它们分别是：

（1）风险“2.1 IGMP 测试范围的变更，由于需求文档中缺少非功能性需求的描述”和“3.2 测试计划没有经过详细的评审”；

（2）风险“2.3 IGMP 测试工作量的偏低估算，有些测试活动没有估算在内”和“1.4

IGMP 测试人员需要参与其他更高级别的测试任务，例如，产品的入网测试”。

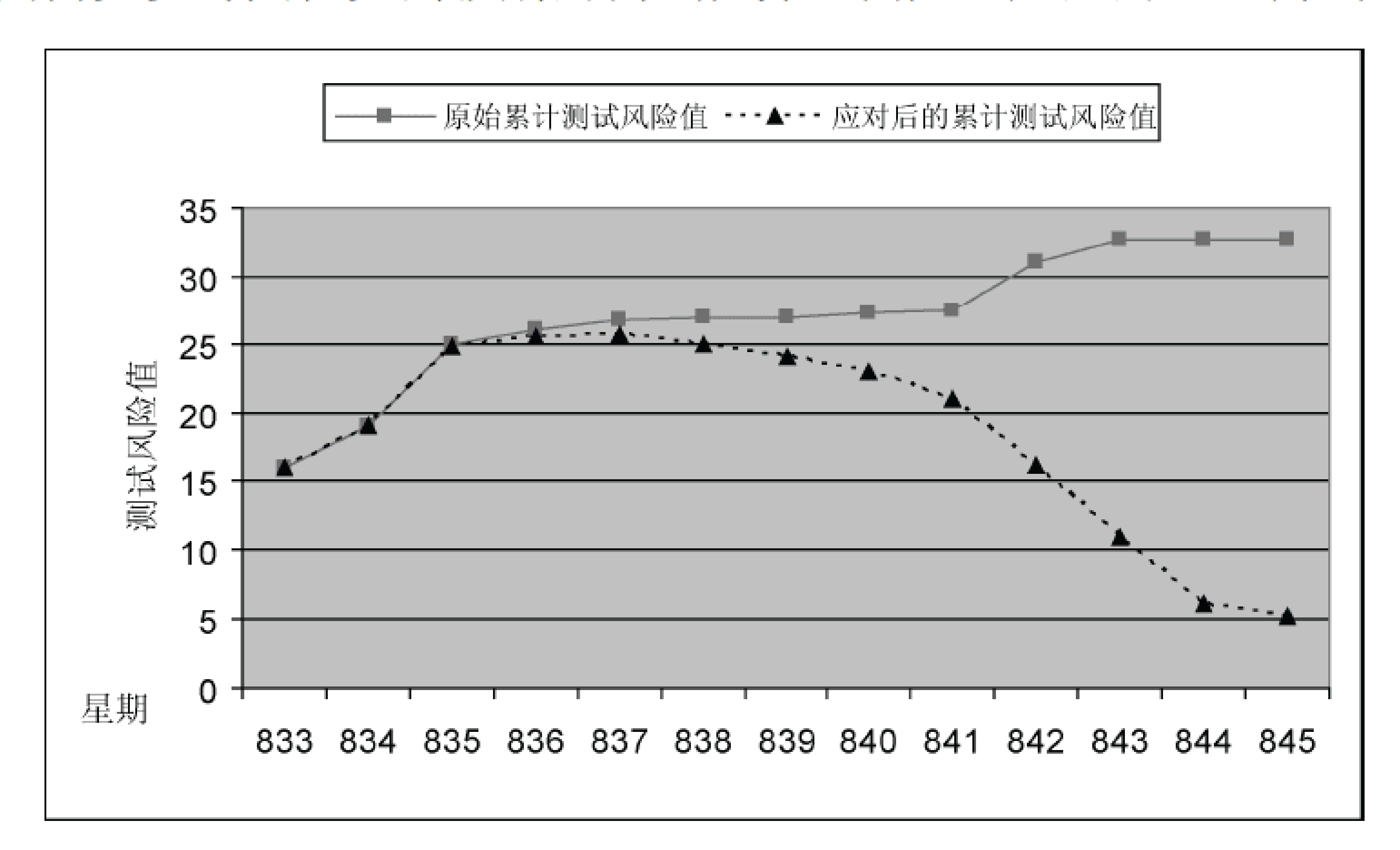


图 2-9 IGMP 测试风险趋势

由于这 4 个测试风险没有在 IGMP 测试活动的早期（测试计划阶段）识别，导致原始累计测试风险值分别在 W836 和 W842 发生了较大的增加。经过总结分析，后续项目可以从下面几点吸取经验教训。

（1）测试风险管理指标（TRMI）只有 75%，和测试计划中要求的 85%有些差距，即在测试计划阶段中遗漏了较多的测试风险，导致在后续的测试阶段不断有新的测试风险被识别。因此，在后续项目的测试风险管理过程中，需要采取更加适合的风险识别方法和邀请更多的项目干系人参与测试风险识别，以提高在测试计划阶段识别测试风险的效率。

（2）在项目早期采取更多的静态测试方法，例如，增加对系统需求规格说明的评审，特别是针对非功能性需求的评审。避免在测试的后期发现严重的非功能性方面的缺陷，影响产品的质量和延迟版本发布时间。

（3）强制项目干系人参与测试计划的评审，特别是在测试范围、测试方法、测试场景、测试网络拓扑等方面的评审，尽量使测试环境符合用户的要求和使用特点。

（4）测试工作量估算偏低的问题，将测试计划估算期间没有覆盖的测试活动和测试任务添加到工作量估算列表中，不断更新工作量估算列表。IGMP 测试工作量的估算并没有考虑测试过程中发现的缺陷的确认测试和相关回归测试工作，导致测试执行进度的延后，并且经常出现测试人员加班加点的情况。

2.6.2 缺陷

缺陷是测试过程监控的重要输入，同时它常常作为测试出口准则的主要条目。与缺陷相关的度量主要包括：

（1）已报告（发现）的缺陷总数与已解决（修复）的缺陷总数之间的比例。

（2）缺陷或缺陷出现率的平均时间间隔。

（3）按下列缺陷分类统计的缺陷数或百分比。

① 特定的测试项或组件。

② 根本原因。

③ 缺陷来源（如需求规格说明、新特性、回归测试等）。

④ 缺陷引入、发现和移除的阶段。

⑤ 优先级和严重程度。

⑥ 拒绝或重复的缺陷报告数目。

（4）从报告缺陷到修复缺陷所花的时间趋势；

（5）由于缺陷修复而引入的新缺陷数目（有时也称子缺陷）。

下面分别从缺陷发现阶段分布、缺陷所属模块分布和发现缺陷的测试类型分布三个方面，阐述缺陷因素在测试监视和控制过程中的作用。

1．案例分析：缺陷发现阶段分布

图 2-10 是某软件产品的基于软件开发生命周期不同阶段发现的缺陷数目统计，通过它可以清晰地了解各个阶段发现的缺陷的数目，分析开发过程和测试过程的有效性。基于阶段的测试缺陷分布由下面几个数据组成。

（1）目标缺陷：项目在各个阶段期望发现的缺陷数目。

（2）实际缺陷：项目在各个阶段实际发现的缺陷数目。

（3）软件开发生命周期不同阶段：该度量数据涉及的阶段包括评审、组件测试、集成测试、系统测试、验收测试和用户反馈。

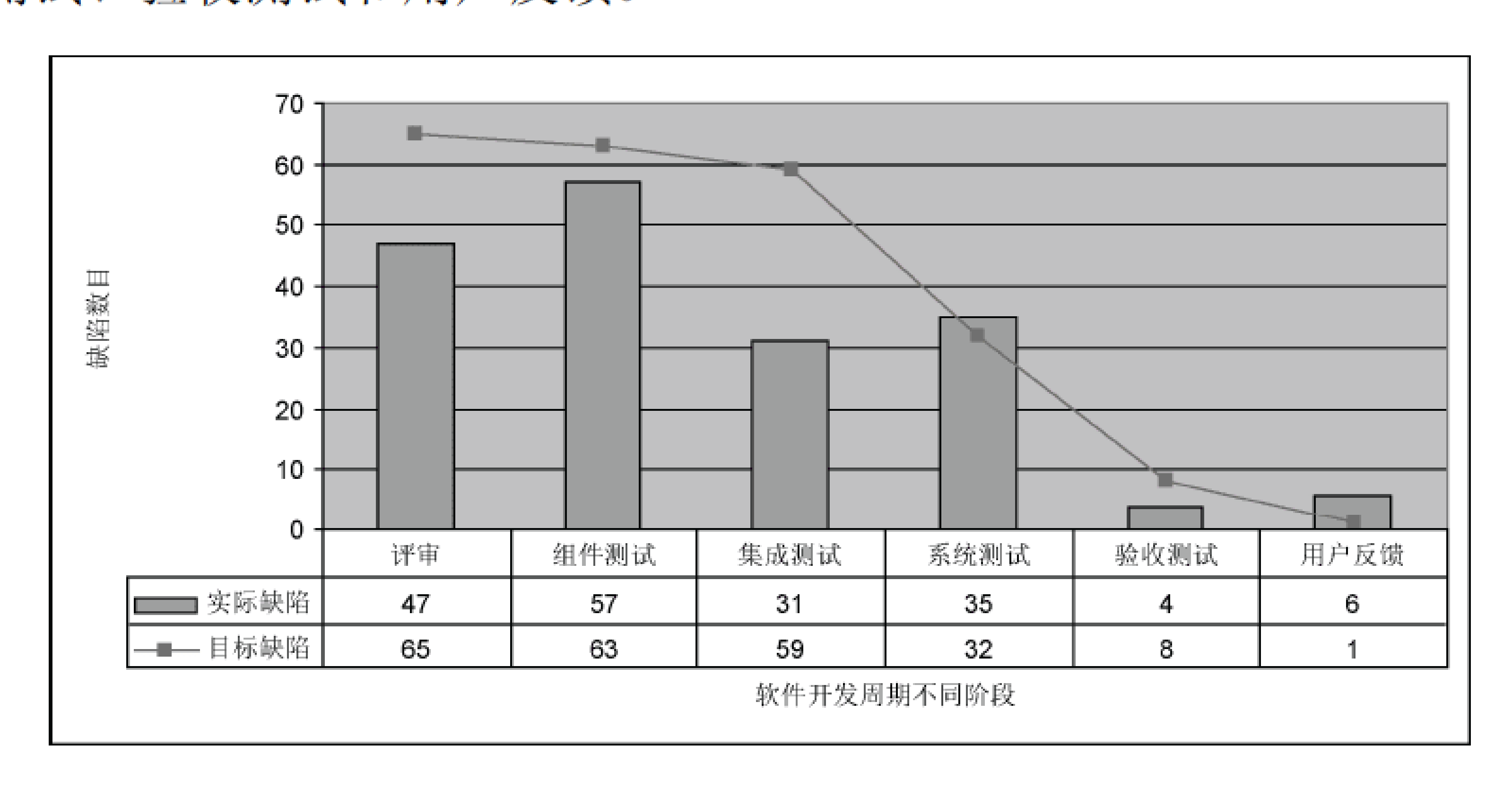


图 2-10　基于阶段的缺陷分布

根据缺陷发现的不同阶段的分布度量，可以通过以下方面分析软件开发生命周期各个阶段发现缺陷的情况。

（1）缺陷在各个阶段的分布：从整体分布上看，该项目通过评审和组件测试发现的缺陷数目超过 50%，说明整个团队在项目前期的投入得到了较好的回报。

（2）评审、组件测试和集成测试过程中发现的缺陷数目都低于期望值，因此，需要分析造成这种结果的原因，例如，输出的软件工作产品质量比较高、开发人员编写的软件代码质量比较好，还是由于测试人员缺乏测试对象相关的经验、知识和技能、测试用例的设计和执行的质量比较差，或者测试的工作量没有得到保证等。

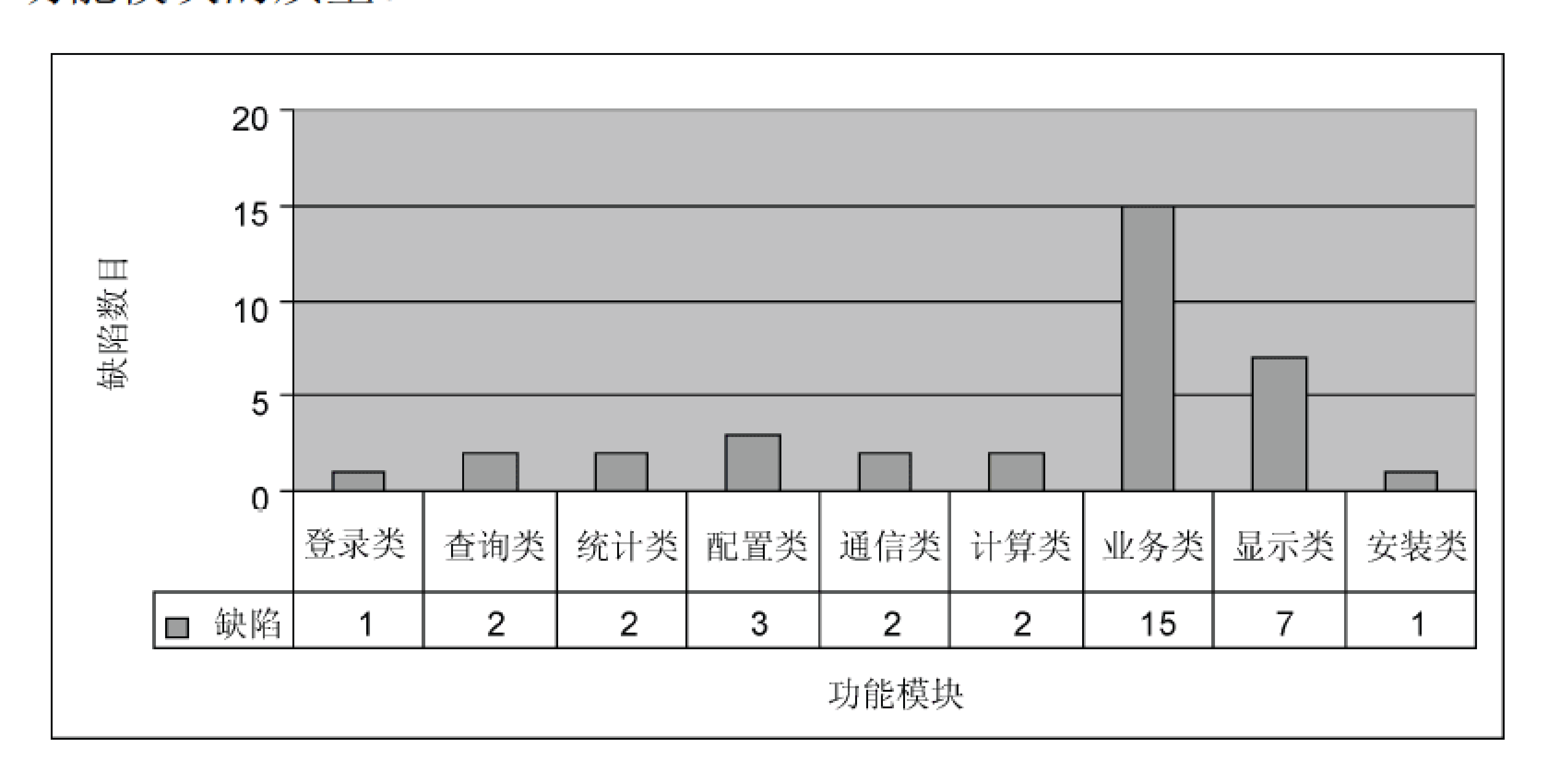
（3）系统测试和用户反馈阶段发现的缺陷数目高于预期值，尤其是用户反馈阶段发现的缺陷数目大大高于预期，说明整个软件质量仍然有待进一步提高。需要针对用户反馈的

缺陷做进一步的分析，以确定开发和测试的改进点。

2．案例分析：缺陷所属模块分布

缺陷按不同功能模块的分布，可以用来表示在某个测试阶段在不同软件功能模块中发现的缺陷数目。通过不同模块发现缺陷的数目的不同，结合执行测试用例数目等数据，分析开发和测试中可能存在的问题，并帮助决定后续开发和测试的重点。图 2-11 是某项目发现的缺陷按模块分布图。

根据图 2-11 所示，“业务类”和“显示类”功能模块发现的缺陷数目明显要高于其他功能模块，需要对这两个模块进行进一步的分析，以确定发现的缺陷数目偏大的原因。对基于不同功能模块中发现的缺陷数目进行分析的时候，如果能与功能模块的规模（例如代码行或功能点），以及针对该功能模块执行的测试用例数目相结合，就可以更加清晰地分析和评估各个功能模块的质量。



	登录类	查询类	统计类	配置类	通信类	计算类	业务类	显示类	安装类
缺陷	1	2	2	3	2	2	15	7	1

图 2-11　基于模块的缺陷分布

“业务类”模块中发现的众多缺陷导致大量相关测试用例被阻塞，项目经理、开发经理和测试经理以及测试人员等需要讨论如何解决这些缺陷以及被阻塞的测试用例，可以采用的方案有：开发团队需要发布一个中间的版本优先解决这些缺陷，从而加快这个模块相关测试用例的执行；按照原来的计划等待下一个软件版本的发布，测试团队继续进行测试；或者测试团队在等待下一个版本的过程中，将测试资源和人力分配到软件的其他模块中继续其他功能模块的测试。

根据如图 2-11 所示的基于模块的缺陷分布，可以发现“业务类”模块和“显示类”模块中发现的缺陷数目特别多。根据测试过程中的缺陷集群效应，这两个功能模块应该是后续测试的重点。同时还需要分析造成在这两个功能模块中存在较多缺陷的根源，例如，需求阶段没有采用合适的评审类型开展静态测试、没有有效地开展组件测试、这两个功能模块采用了新的技术、参与这两个功能模块的开发人员在技能方面有差距（由于是新员工）等。分析结果可以帮助改进软件开发过程和测试过程，以提高后续项目开发的效率和软件产品的质量。

3．案例分析：发现缺陷的测试类型分布

缺陷按测试类型分布可以用来表示在某个测试阶段，针对软件进行的不同测试类型发现的缺陷数目。根据发现缺陷的测试类型的分布，结合执行的测试用例数目等数据，可以

分析开发过程和测试过程中可能存在的问题，并帮助决定后续开发和测试的重点。图 2-12 是某项目基于测试类型的缺陷分布图。

如图 2-12 所示，每个测试类型都有缺陷发现，一定程度上说明了测试用例设计的质量比较高，设计的测试用例不仅覆盖了功能性测试用例，也包括要求的非功能性测试用例。其中“压力”测试类型发现了 7 个缺陷，仅次于“功能性”测试类型的 9 个缺陷，说明该系统在压力测试方面的表现较差。而压力测试暴露的问题通常都难以解决，因为它和整个软件系统的设计、架构或使用的硬件密切相关。因此，需要针对“压力”测试类型中发现的较多缺陷这个问题进行分析，例如，该项目计划阶段的风险识别活动中遗漏了系统设计架构方面的风险，没有在项目早期及时采取合适的应对措施降低该风险。

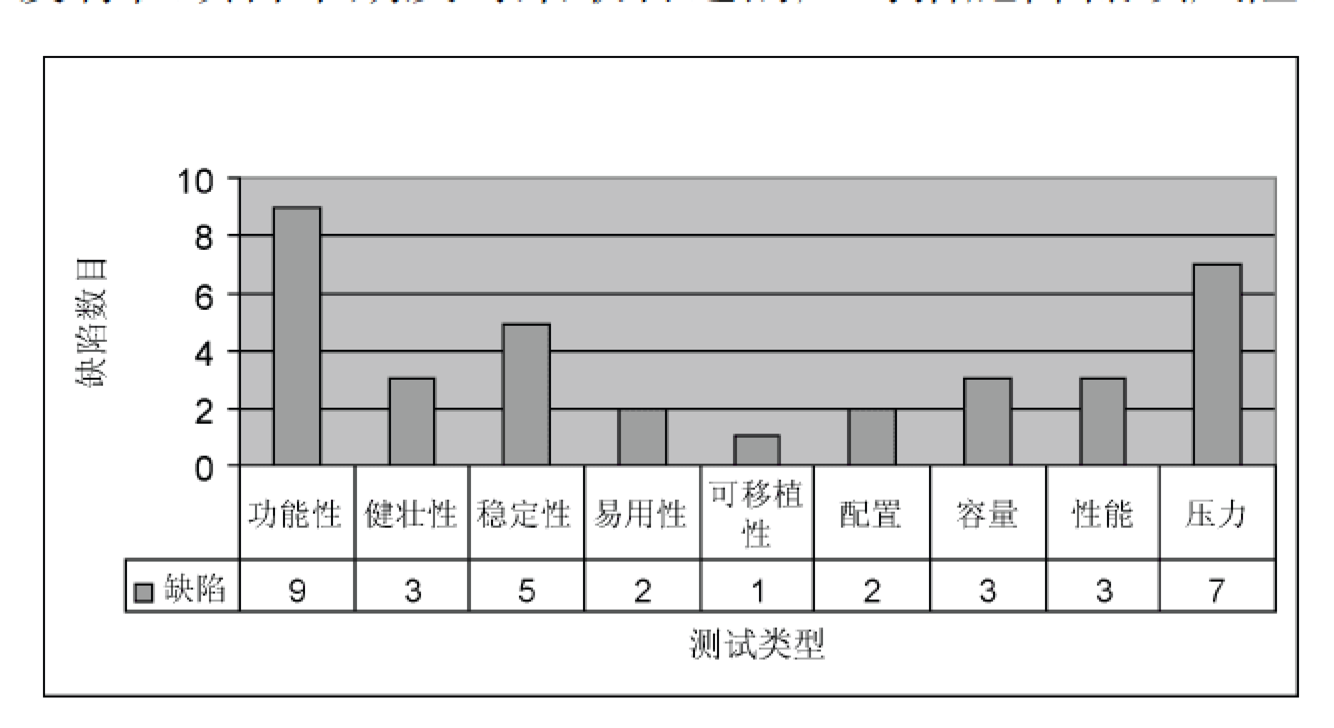


图 2-12　基于测试类型的缺陷分布

2.6.3　测试

前面提到的风险状态和趋势度量信息，更多的是从过程改进方面阐述的，也就是如何通过对风险状态和趋势的监控，为将来的风险识别和应对提供一些经验和教训，从而提高风险管理的成熟度，例如，不断完善风险检查表。而测试进度进展监控的第三个维度与测试相关，其主要包括的度量如下。

（1）不同测试状态的测试数目：已计划的、已详细说明（已实施）的、已运行的、执行通过的、执行失败的、无法执行的和跳过不执行的等。

（2）回归测试和确认测试的状态，包括整体趋势和未通过的回归测试总数及执行未通过的确认测试总数。

（3）计划的每日测试时间与实际的每日测试时间之间的比例。

（4）测试环境的可用性（测试团队准备的测试环境可用时间占计划测试时长的百分比）。

下面以测试用例设计进度和测试用例执行进度为例，阐述测试数据信息在测试过程监控中的作用，以及如何为将来的项目提供经验教训。

1．案例分析：测试用例设计进度

图 2-13 显示了某软件项目的系统测试用例设计进度，主要用来控制测试设计是否按照计划的时间进度要求进行。测试用例设计进度图中主要由下面几个数据组成。

（1）总的数目：总的测试用例数目，是根据估算得到的需要新设计的测试用例数目。由于估算会随着测试设计的进展越来越准确，所以，总的测试用例数目可能会随着时间发生变化。

（2）原来计划：根据估算得到的总的测试用例数目和组织定义的测试用例设计速率，以及测试人员的数量，对测试用例设计的进度进行计划。

（3）实际计划：由于测试用例估算存在偏差，所以需要根据测试用例估算的变化和实际的测试资源的变化，调整测试用例设计的进度计划。

（4）实际设计：是测试人员在设计过程中实际输出的测试用例数目。

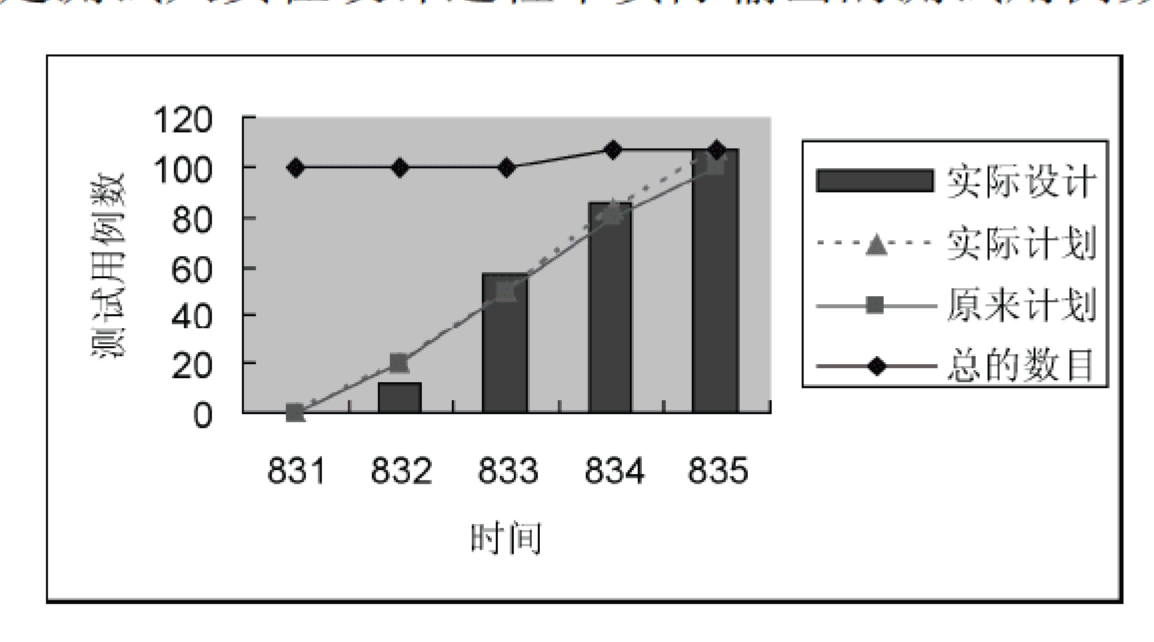


图 2-13　系统测试用例设计进度

根据图 2-13 中提供的信息，可以从以下方面分析和评估测试用例设计进度。

（1）在第 834 周，总的测试用例数目从原来的 100 个变成了 107 个，即实际需要设计的测试用例数目比原来估算的多了 7 个。按照原来的计划是第 835 周完成全部的测试用例设计，但是由于在第 834 周增加了 7 个测试用例，根据项目的测试资源和测试人员的安排，及时完成测试用例的设计存在较大的风险。因此，测试经理需要根据当前的测试用例设计状态采取合适的应对措施，使得测试用例设计进度回到原来计划的时间进度。测试经理和项目经理沟通之后，项目经理认为无法更改原来的测试用例设计时间进度计划。所以，测试经理经过与测试设计人员的沟通，通过加班的方式，使得测试用例设计按时在第 835 周完成。

（2）其次，需要分析为什么在第 834 周总的测试用例数目会从原来的 100 个变为 107 个，并将分析的结果作为经验教训指导后续的测试活动和项目。通过分析之后发现，由于该功能的测试用例数目估算主要是基于系统需求规格说明展开的，而在测试用例设计过程中发现，系统需求规格说明中主要定义了功能性需求，而对非功能性需求定义的较少。而测试用例中需要覆盖组织层面定义的各种测试类型，这样就出现了有些测试类型无法从需求规格说明中得到的问题。那么，为什么在项目的早期没有发现非功能性需求遗漏问题呢？作为经验教训，测试人员应该在系统需求阶段就参与相关文档的评审，并从测试的角度来提出建议，避免非功能性需求的遗漏。

2．案例分析：测试用例设执行进度

图 2-14 显示了某项目系统测试用例执行进度，主要用来控制测试执行是否按照测试计划的要求进行，是否能够顺利完成测试执行进度。测试用例执行进度主要由下面几个数据组成。

（1）总的数目：总的测试用例数目，是根据估算得到的需要执行的测试用例数目，包括新设计的测试用例数目、相关的回归测试用例数目。由于回归测试用例的数目可能会随着测试用例的执行而发生变化，例如，由于在某个模块发现的缺陷比预计的多，就可能要求增加这个模块的回归测试用例数目。因此，随着测试用例执行的进展，总的测试用例数目可能会发生变化。

（2）原来计划：根据估算得到的总的测试用例数目和组织定义的测试用例执行的速率，以及测试人员的数量，对测试用例执行的进度进行计划。

（3）实际计划：由于选择的回归测试用例数目可能会发生变化，所以需要根据回归测试用例数目的变化，调整测试用例执行的进度计划。

（4）通过的数目：在测试执行过程中，测试用例执行得到的结果和预期结果相同，或者符合预期，认为测试用例执行通过。

（5）失败的数目：在测试执行过程中，测试用例执行得到的结果和预期结果不一样，或者不符合测试人员的预期。经过仔细检查和确认之后，认为是由于测试对象中存在的问题引起时，就认为该测试用例执行结果为失败。此时需要提交缺陷报告，以跟踪缺陷的修复和验证。

（6）被阻塞的数目：由于测试仪表欠缺、相关的缺陷没有修改等原因，导致目前无法执行的测试用例数目。

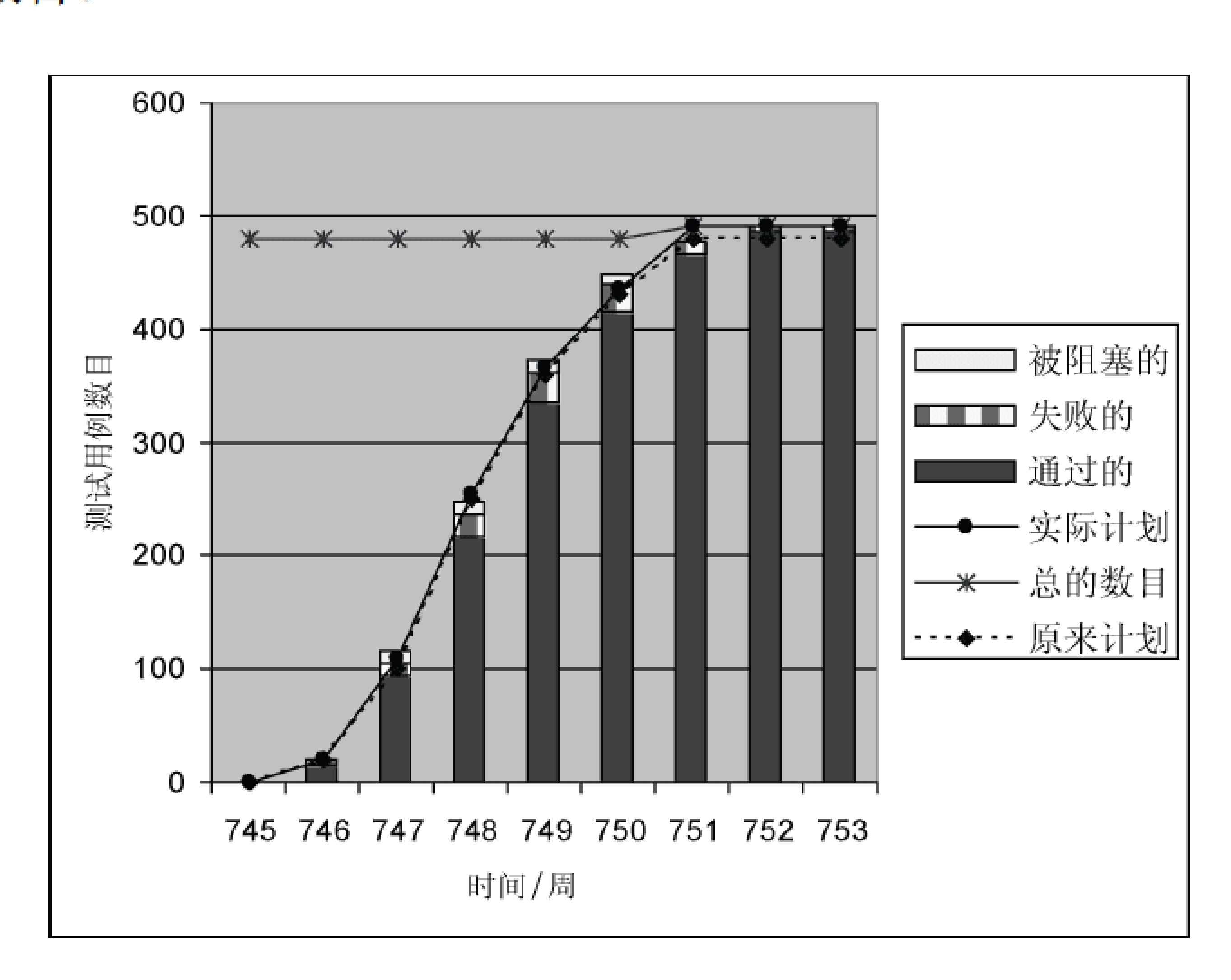


图 2-14　系统测试用例执行进度

根据图 2-14 提供的数据和曲线，可以从以下方面分析和评估测试用例执行进度。

（1）在第 751 周，需要执行的测试用例数目从原来计划的 480 个变为 490 个，也就是说，实际需要执行的测试用例数目比原来估算的多了 10 个。需要分析是什么原因导致执行的测试用例数目增加了 10 个。通过分析之后得知：执行过程中发现某个功能模块发现的缺陷比原来预计的多，于是，测试经理为了降低该模块可能存在的风险，针对该模块额外增加了 10 个测试用例。

（2）由于在第 751 周需要执行的测试用例数目增加了 10 个，按照原来的测试用例执

行进度计划在第 751 周完成所有的执行任务，从目前的测试用例执行状态来看，存在的困难比较大，原因如下。

① 首先，比原来计划的测试用例数目多了 10 个，完成 10 个增加的测试用例执行需要额外的测试时间和测试资源。

② 其次，在第 751 周有 25 个状态为失败的测试用例，即针对这 25 个测试用例，后续需要增加测试工作量进行相关缺陷的验证，以及进行确认测试和回归测试。

③ 第三，在第 751 周还有 10 个由于测试仪表缺乏导致被阻塞的测试用例。

（3）由于存在可能会导致测试用例执行延期的上面几个风险，测试经理和项目经理进行沟通之后，从以下两个方面入手应对这些风险。

① 第一，从项目的层面，测试用例执行延期一周。

② 第二，测试经理通过在不同项目之间人员的协调，从其他项目组抽调了一名经验丰富的测试人员，临时加入该项目的系统测试。

2.6.4 覆盖率

测试覆盖率是测试过程中进行监视和控制的主要信息来源，它常常也作为定义软件测试出口准则的重要条目。因此，测试覆盖率的监控不仅可以帮助测试过程监控的实施，也有助于分析和评估测试出口准则的满足与否。和测试覆盖率相关的度量主要包括：

（1）需求和设计要素的覆盖率；

（2）风险覆盖率；

（3）环境和配置覆盖率；

（4）代码覆盖率。

下面从需求的测试覆盖率和平台的测试覆盖率两个方面进行分析。

1．案例分析：基于需求的测试覆盖率

图 2-15 显示了某项目系统测试的基于需求的测试覆盖率。通过检查基于需求的测试覆盖率在测试执行过程中的变化，以及测试用例执行进度等数据，测试经理可以分析测试是否能够顺利完成，以及测试覆盖率是否达到了测试计划中定义的要求（例如，是否满足测试出口准则的覆盖率条目要求）。基于需求的覆盖率图中主要由下面几个数据组成。

（1）原来计划：在测试用例执行过程中，原来计划执行的测试用例在需求覆盖率方面的变化。

（2）实际计划：由于测试执行方面碰到的问题和可能存在的风险，测试经理会根据测试的状态和结果调整测试执行在需求覆盖率方面的要求。

（3）实际覆盖率：执行的测试用例实际达到的需求覆盖率。

基于需求的测试覆盖率定义如下。

$$\text{基于需求的测试覆盖率}=(\text{实际覆盖的需求数目}/\text{总的需求数目})\times 100\%$$

根据图 2-15 提供的数据和曲线，可以从以下方面分析和评估测试用例执行在需求覆盖率方面的变化。

在第 818 周，实际计划的基于需求的测试覆盖率从原来计划的 100%变为 96%。分析基于需求的测试覆盖率在测试过程中发生变化的原因。

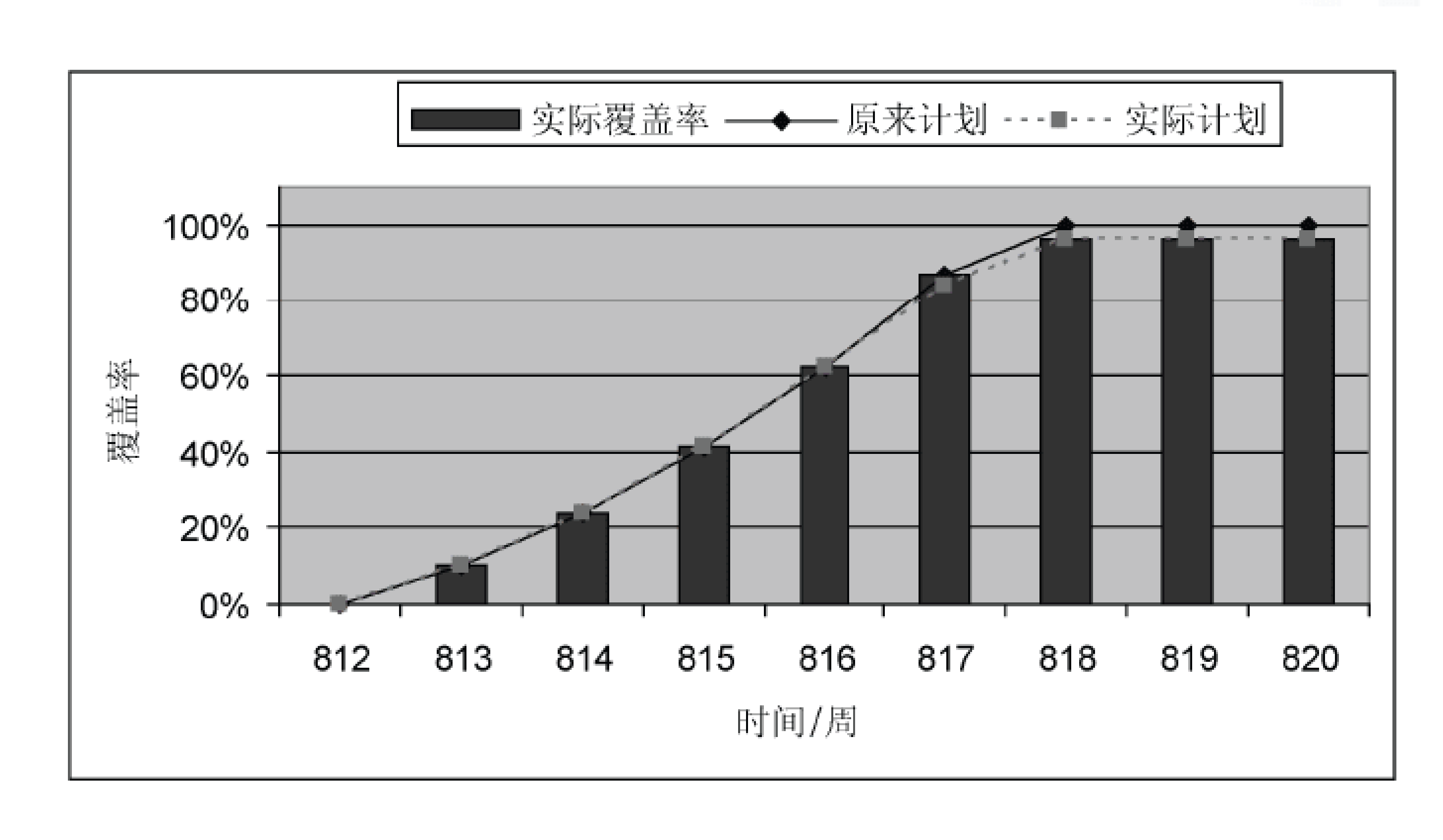


图 2-15　基于需求的测试覆盖率

（1）测试人员由于在项目前期没有预留时间对输出的系统需求规格说明期进行评审，因此，没有发现需求规格说明中遗漏的非功能性需求，导致在测试执行的后期，测试人员不得不增加相应的非功能性测试用例数目。在增加的非功能性测试用例中，其中两个测试用例无法在当前的资源下进行验证：业务的平均失效时间和大量用户同时进行业务访问。因此，在测试执行的后期只能将这两个测试用例置于被阻塞状态，从而导致测试覆盖率的下降。

（2）由于测试人员没有通过评审发现系统需求规格说明中非功能性需求的遗漏，导致在测试计划阶段的风险识别过程中没有识别出针对这些非功能性需求的风险。因此，针对大量用户同时进行业务访问这样的测试用例，由于需要特殊的测试仪器，导致通过人工方式在当前的测试环境中无法对大量用户模拟。

（3）在测试计划中，定义的测试出口准则是要求达到需求覆盖率为 100%，但是由于实际的测试情况已经无法满足这个要求。因此，测试经理和项目经理以及其他项目利益干系人进行沟通和协商之后，只能更新测试计划以反映这个状态，将测试出口准则中关于需求覆盖率的要求从 100%修改为 95%。

通过人工的方式进行统计和跟踪基于需求的测试覆盖率，不仅费时费力，而且效率低下，容易出错。因此，应尽量通过工具的支持对基于需求的测试覆盖率进行统计，并以合理的方式表示，例如，图表方式。

2．案例分析：基于平台的测试覆盖率

基于平台的测试覆盖率，指的是测试用例的执行应该覆盖哪些平台，这里的平台指的是测试对象可能运行的硬件、软件、操作系统、接入系统、服务器端系统等组成的一个综合体。对于 iBAS R1.0 项目的 IGMP 系统测试，由于是该产品的第一个版本，所以可以支持的平台只有两个：IGMP_HARDWARE1 和 IGMP_HARDWARE2。因此，在选择 IGMP 测试用例和相关回归测试用例的时候，很容易覆盖这两个平台，而且很容易实现 100%的基于平台的测试覆盖率。

尽管针对 iBAS R1.0 项目的 IGMP 系统测试实施 100%的基于平台的测试覆盖率是比较简单的。但是在测试实践过程中，如何在有限的时间和资源下，将测试用例有效地覆盖到不同的测试平台上（即测试用例和测试平台之间的多种组合），经常会面临可能的测试平台太多这样的问题，例如，软件产品针对美国和欧洲的两个标准；软件产品可能采用的接入

手段多种多样（ADSL 接入、ADSL2 接入、以太网接入、VDSL 接入等）；软件产品可能采用的硬件类型是不一样的。假如将所有的因素进行组合，得到的测试平台的数目将是非常庞大的，将测试用例完全覆盖所有的测试平台，在时间、成本、资源等方面都是不允许的。因此，测试经理在面临这样的问题时，需要采取合适的策略来平衡测试工作量、测试质量和测试效率，以下是经常采用的策略和建议。

（1）将测试用例映射到不同测试平台的时候，首先考虑用户最常用的运行平台。对于 iBAS 产品，它们会部署在不同国家和地区的不同运营商。因此，测试经理应该和产品经理、市场技术人员进行沟通和讨论，列出不同运营商采用的运行平台，按照一定的标准（例如，根据不同平台在用户中使用的容量）对测试平台划分不同的优先级，然后根据不同的优先级确定测试用例在不同测试平台的分布比例和数目。

（2）根据系统的设计规格说明中定义的不同硬件类型的属性、特点和它们之间的关系，例如，有些硬件是有继承关系的（如母板和子板之间的关系），确定测试用例在平台上的映射分布。对于母板和子板这样的硬件类型，一般而言，测试用例会更多地映射在子板这样的硬件类型上，而对于母板硬件类型上的测试用例分布，在工作量和资源紧张的时候，可以适当地降低其测试优先级。

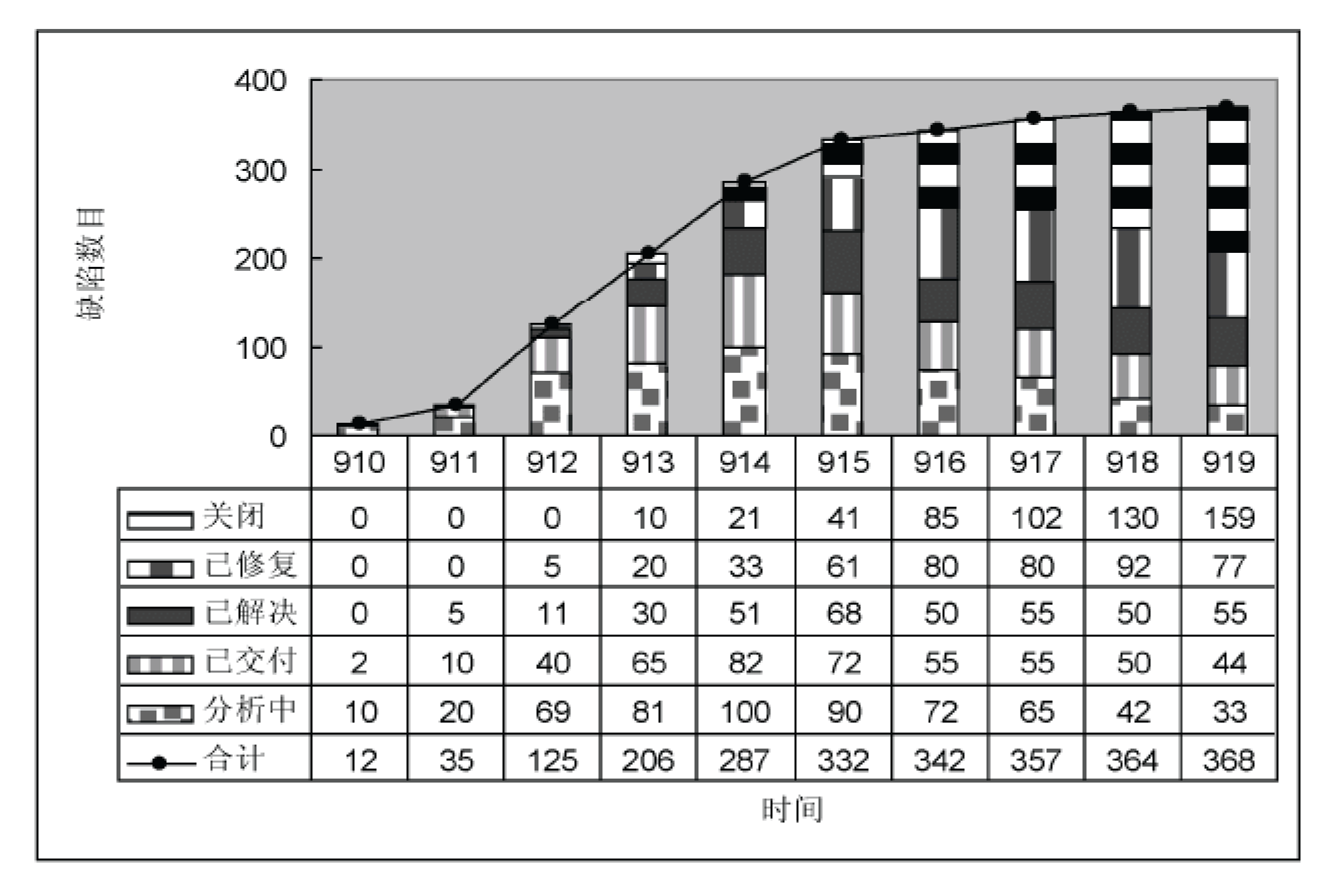
（3）测试人员需要在开发人员和系统人员的帮助下，分析执行的测试用例是否和测试平台中的硬件类型相关。对于测试对象功能和硬件类型无关的测试用例，没有必要将它们平均分布在不同的硬件类型上，而是选择性地在不同的硬件类型上测试即可。

2.6.5 信心

信心一方面来自测试过程中发现缺陷的状态演变趋势和测试通过率的趋势，另一方面来自测试人员对测试对象的主观印象。测试过程监控也可以基于测试人员对测试对象的信心展开。下面对测试发现缺陷状态变化和测试用例通过率两个指标进行描述，并阐述它们在测试监视和控制过程中的作用。

1．案例分析：测试发现缺陷状态变化

图 2-16 是某项目的测试发现缺陷状态变化图，该图是基于每个星期发现的缺陷。



	910	911	912	913	914	915	916	917	918	919
关闭	0	0	0	10	21	41	85	102	130	159
已修复	0	0	5	20	33	61	80	80	92	77
已解决	0	5	11	30	51	68	50	55	50	55
已交付	2	10	40	65	82	72	55	55	50	44
分析中	10	20	69	81	100	90	72	65	42	33
合计	12	35	125	206	287	332	342	357	364	368

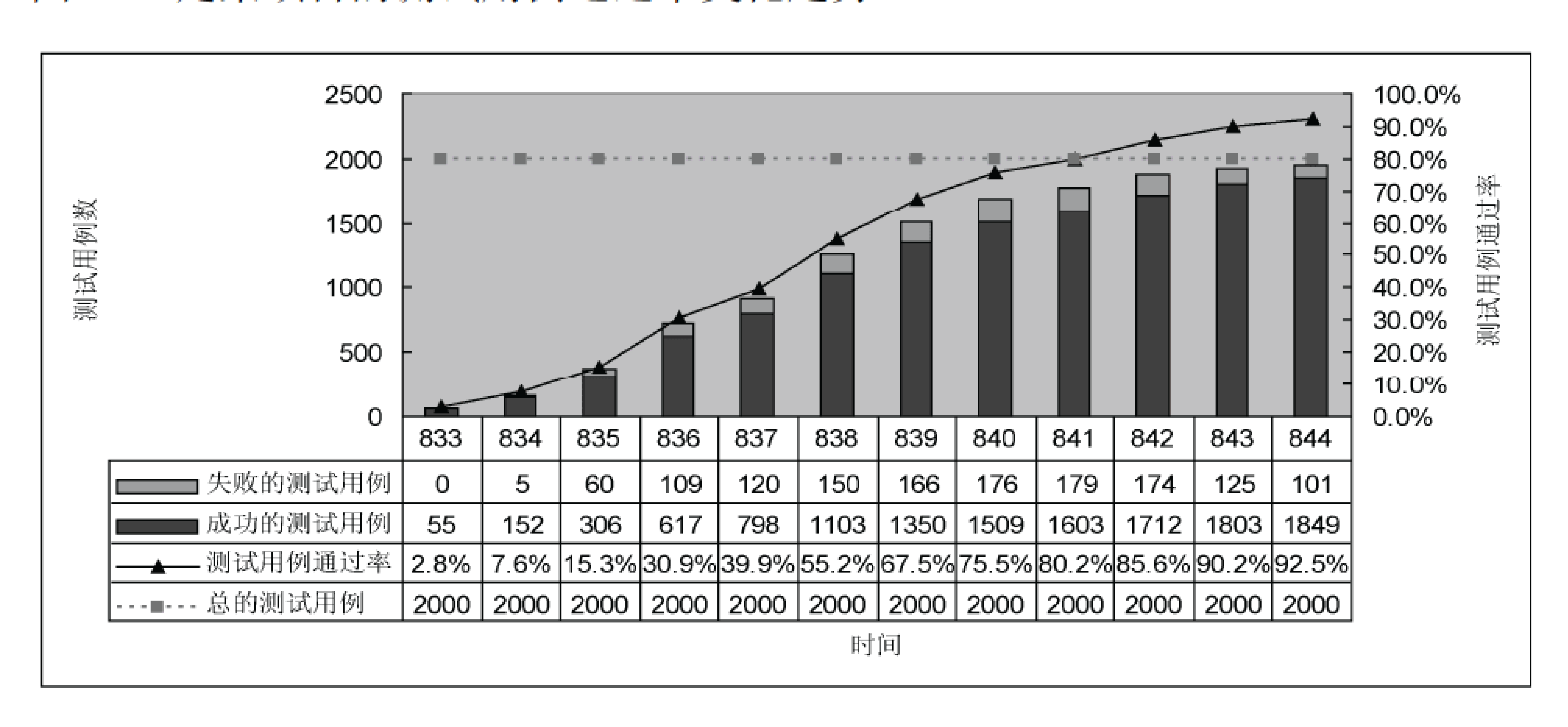
图 2-16 测试发现缺陷状态变化图

如图 2-16 所示，该项目在测试执行的前两周，测试人员发现的缺陷比较少，可能的原因是由于测试人员不熟悉测试环境和被测试对象。随着测试人员对测试环境和测试对象的逐步了解，大量的缺陷被迅速发现，在第 912、913、914 和 915 周共发现了 297 个缺陷，远远高于其他时间。但是将发现的缺陷数目和关闭的缺陷数目进行对比发现，尽管大量的缺陷被发现，但是开发人员并没有及时对这些缺陷进行处理，在测试执行的前三周，发现的缺陷有 125 个，但已修复和解决的缺陷只有 16 个（已修复的 5 个和已解决的 11 个），说明开发人员在修复缺陷方面投入严重不足。项目经理应该对此引起高度重视，保证开发人员的投入，否则将严重影响后续的测试任务和产品质量，甚至导致产品无法按时提交给客户。

另外，整个缺陷的发现趋势随着测试执行的进行呈现逐渐收敛的趋势，从第 915 周开始，每周发现的缺陷都比较少。但是发现的缺陷数目呈现逐渐收敛的趋势，并不能完全说明测试对象的质量已经趋于稳定，以及认为可以结束测试执行活动。通过分析图 2-16 缺陷的不同状态分布可以发现，尽管发现的缺陷数目增加的并不多，但是仍然有大量的缺陷没有关闭，这个时候，首先需要分析其中的原因，例如，是否是由于测试人员在缺陷验证方面投入不足。其次，需要重新评估测试执行活动，判断是否能够按时完成测试执行的任务。从图中可以发现，由于存在较多的没有验证的缺陷，导致后续的测试任务繁重：缺陷的确认测试和相关的回归测试任务等。因此，测试经理需要和测试人员进行有效沟通，及早采取应对措施，避免由于这些缺陷而导致的测试执行延期风险。

2．案例分析：测试用例通过率

图 2-17 是某项目的测试用例通过率变化趋势。



	833	834	835	836	837	838	839	840	841	842	843	844
失败的测试用例	0	5	60	109	120	150	166	176	179	174	125	101
成功的测试用例	55	152	306	617	798	1103	1350	1509	1603	1712	1803	1849
测试用例通过率	2.8%	7.6%	15.3%	30.9%	39.9%	55.2%	67.5%	75.5%	80.2%	85.6%	90.2%	92.5%
总的测试用例	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000

图 2-17　测试用例通过趋势图

如图 2-17 所示，整体测试执行进度良好，通过的测试用例数稳定增加，说明整个测试环境和测试团队比较稳定。到第 844 周为止，整个测试用例通过率达到 92.5%，说明系统的整体质量良好。但是仍然还有 101 个测试用例执行没有通过，存在一定程度的风险，需要对这些没有通过的测试用例进行进一步的分析。造成测试用例未通过的原因可能有：测试环境没有到位、产品存在缺陷或测试用例本身设计的问题等。

测试过程监控的输入有时候也来自测试人员对测试对象质量的主观信心，例如，测试经理通过和相关测试人员的访谈，根据测试人员在测试过程中对测试对象的功能模块的了

解程度和表现行为，获得他们对测试对象质量的评判。

度量除了可以按照上述测试进度监控的 5 个维度进行分类，也可以与基本测试过程中的每个阶段进行关联。测试经理可以按照项目目标和测试过程要求，采用度量数据监督测试过程的每个阶段，以及评估是否达到项目目标的进展。

和监控测试计划和控制活动相关的度量包括：

（1）风险、需求和其他测试依据要素的覆盖率；

（2）缺陷发现情况；

（3）计划开发测试件和执行测试用例的时长与实际时长之间的比例。

和监控测试分析活动相关的度量包括：

（1）识别的测试条件数目（如根据测试依据识别的产品风险或其他测试条件）；

（2）测试分析过程中发现的缺陷数。

和监控测试设计活动相关的度量包括：

（1）测试用例覆盖的测试条件百分比（如根据测试条件开发的测试用例）；

（2）测试设计过程中发现的缺陷数。

和监控测试实施活动相关的度量包括：

（1）测试环境配置的百分比；

（2）测试数据记录加载的百分比；

（3）测试用例自动化的百分比。

和监控测试执行活动相关的度量包括：

（1）已运行、执行通过和失败的测试占已计划的测试的百分比；

（2）已运行或执行通过的测试用例覆盖的测试条件的百分比；

（3）计划与实际报告或解决的缺陷数目之间的比例；

（4）计划与实际达到覆盖率之间的比例。

监控测试进度和测试完成活动的度量包括里程碑、入口准则和出口准则（测试计划中定义和批准的）的映射，其中可能包括以下内容。

（1）计划的测试条件、测试用例或测试规格说明的数目，以及按照测试是否通过，分别统计已经执行通过的和失败的测试用例数目；

（2）按照严重程度、优先级、当前状态、受影响的子系统或其他分类方式，统计总的缺陷数目（见第 4 章）；

（3）要求的、接受的、开展的和已测试的变更数目；

（4）计划成本与实际成本之间的对比；

（5）计划工期与实际工期之间的对比；

（6）测试里程碑的计划日期与实际日期之间的对比；

（7）有关测试的项目里程碑（如代码冻结）的计划日期与实际日期之间的对比；

（8）按照已缓解与未缓解的风险、主要的风险区域、测试分析后发现的新风险等分类统计产品（质量）风险状态；

（9）由于阻塞事件或计划的变更导致的测试工作量、成本或时间损失的百分比；

（10）确认测试和回归测试状态。

和监控测试结束活动相关的度量包括：

（1）测试执行期间已执行的、通过的、失败的、无法执行的和已计划但没有执行的测试用例的百分比；

（2）纳入可复用的测试用例库的测试用例的百分比；

（3）自动化测试用例的百分比或计划的与实际的自动化测试用例百分比对比；

（4）集成到回归测试的测试用例的百分比；

（5）已解决或未解决的缺陷报告的百分比；

（6）识别和归档的测试工作产品的百分比。

另外，标准的项目管理技术，如工作分解结构，通常会用来监视测试过程。敏捷团队中的测试活动，经常作为燃尽图上用户故事进展的一部分。而精益管理技术中基于用户故事的测试进度，通常是通过将用户故事卡从看板的不同列之间移动来实施监视的。

测试过程中确定一组度量标准后，度量数据可以通过口头陈述、表格数值或图形等方式来进行汇报。度量数据可以有如下多种用途。

（1）分析：根据测试结果找出可观察到的趋势和原因。

（2）汇报：将测试结果与感兴趣的项目干系人进行沟通。

（3）控制：根据度量数据相应变更整个测试或项目活动，以及监控过程应对的结果。

测试度量数据的收集、分析和报告的具体方式，取决于具体的信息需要、目标和使用这些度量数据的个人能力。另外，测试报告中提供的具体信息，也应该根据不同的利益干系人要求而做出调整。

为了达到测试控制的目的，测试过程中提供给测试经理的度量数据（测试计划完成之后），应该能够引导测试工作成功地完成测试使命、测试策略和测试目标。因此，测试规划必须考虑度量信息的要求，测试监视必须包括收集任何需要的工作产品度量。需要的信息量和采集信息需要的工作量取决于各种项目因素，包括项目规模、复杂度和风险。

测试控制一定要与测试过程中生成的信息和软件项目当前不断变化的环境相适应。例如，假如动态测试在某些认为不可能有很多缺陷的区域发现了缺陷群，或者由于测试开始时间延迟导致测试执行周期缩短，则必须对风险分析和测试计划做出修正。从而可能需要对测试优先级重新设定，或者对剩余的测试执行工作量重新进行分配。

假如测试进展报告中的度量数据与测试计划中的要求出现了偏差，则测试经理需要实施测试控制活动。测试控制的目的是为软件项目或者测试修正方向，以更好地达到测试计划中制定的目标。根据测试结果在软件项目控制工作量上施加影响时，需要考虑以下应对措施。

（1）修改产品风险分析、测试优先级或测试计划；

（2）增加测试资源或增加项目或测试工作量；

（3）推迟软件产品发布日期；

（4）放松或加强测试出口准则；

（5）改变项目的范围（功能或非功能）。

测试经理在实施上述测试控制应对措施时，需要确保在项目或者业务利益干系人之间达成共识，并且取得软件项目管理人员的同意。

不同的软件项目利益干系人，对测试报告中发布的信息的要求不同，例如，项目管理人员与业务管理人员的要求也不一样。因此，测试经理提供测试报告信息时，根据对象的

不同需要进行合理的裁剪，例如，项目经理最可能感兴趣的是关于缺陷的详细信息，而业务经理最关注的可能是产品风险的状态。

2.7 测试的商业价值

虽然大多数组织认为测试在某种意义上是有价值的，但很少有管理人员（包括测试经理）可以以量化的方式描述或明确界定测试的价值。另外，测试经理和测试人员有时候过于关注测试技术的细节，例如，各个测试级别的具体活动，却往往忽略了与测试相关的更重要的一个问题：测试的商业价值。这些问题应该是项目利益干系人，特别是测试经理所应该关注的。测试的商业价值可以分为定量的价值和定性的价值。

（1）定量的价值：包括发现缺陷并在产品发布前定位或修复这些缺陷；发现缺陷并了解在产品发布前依旧存在的缺陷（发现但由于各种原因没有修改的缺陷，也可能提供了针对缺陷的补救措施）；通过测试减少风险并发布有关项目、过程和产品状态的信息。

（2）定性的价值：包括提高软件产品质量的声誉；使软件产品发布更顺利和更可预测；增强和建立对软件产品的信心；降低软件产品功能失效甚至造成人员伤亡的可能性，避免承担法律责任。

质量成本分析（有时候称为不良质量成本）是进行测试商业价值分析的非常有效的手段。测试经理在面对诸如为什么进行测试、测试的价值在哪里、什么时候可以结束测试等问题时，可通过对质量成本进行分析和量化，说明测试的商业价值。

质量成本指的是，为了保障软件产品质量而开展的活动以及为解决问题所花费的总成本，包括预防、检测和修复缺陷的相关工作需要花费的成本。质量成本非常巨大，可能占到销售额的 20%～40%[①]。测试过程中有些质量成本是可以大大降低的，甚至是可以避免的。质量成本分析是用于度量测试的价值和效率的成熟方法。不良质量的例子有代码错误、设计错误、用户手册错误，以及维护性差的复杂代码等。

通常情况下，质量成本可以分为 4 个类别[②]：预防成本、检测成本、内部失效成本、外部失效成本。

2.7.1 预防成本

预防成本指的是用来预防不良质量的活动成本。预防成本是组织避免软件产品开发过程或者避免软件产品交付使用后出现各式各样缺陷的活动成本，也可以是用来减少过程本身相关缺陷的成本。与预防成本相关的主要活动如下。

（1）培训：例如，通过培训可以提高开发人员的技能，从而可以减少在软件工作产品中引入缺陷。培训费用是预防成本的重要组成部分，这里包括培训资料的准备、培训讲师、场地和各种培训设备的费用，同时参加培训所花费的时间成本，也包括在这里面。

---

① Gryna, F M．“Quality Costs” in Juran, J M & Gryna, F M (1988, 4th Ed.), Juran’s Quality Control Handbook, McGraw-Hill.

② 来自于 Cem Kaner 的 The Law of software quality。

（2）制定和推广开发过程：好的开发过程能够有效地预防缺陷。明确定义开发过程的各项活动、输入域输出、需要的技能、使用的工具和编程规范等，能够更好地预防缺陷的发生。开发过程的制定、推广和更新都需要成本。

（3）早期原型设计：通过原型的方式，可以保证软件产品设计的可行性，也可以尽早与客户进行沟通，保证开发的软件产品能够满足客户的需求。

2.7.2　检测成本

检测成本指的是用来发现软件产品中质量问题的活动成本。检测成本相关的活动能够检测软件系统、内部组件或者开发过程中存在的问题，缺陷修复之后可以减少软件产品发布后遗留到用户的缺陷数目。

检测成本和预防成本的区别在于，检测成本以发现缺陷为目的，而预防成本以避免引入缺陷为目的。开发过程中的很多活动同时兼有检测成本和预防成本的特点，例如，针对设计规格说明的评审，它既可以属于预防成本，也可以属于检测成本。假如评审设计规格说明的目的是为了查找错误和缺陷，那么它属于检测成本；假如评审设计规格说明的目的是为了查找一些方法来加强设计能力，那么它属于预防成本。和检测成本相关的主要活动有：

（1）代码走查；
（2）组件测试；
（3）集成测试；
（4）系统测试；
（5）验收测试；
（6）规格说明评审。

2.7.3　内部失效成本

内部失效成本指的是在软件产品交付客户之前产生的失效而导致的成本，即针对开发过程中发现的错误、缺陷或者失效而采取相应的应对措施所需要的成本，例如，缺陷的修复成本等。和内部失效成本相关的主要活动有：

（1）各种开发和测试活动的返工；
（2）缺陷修复；
（3）确认测试和回归测试（由于修改缺陷而导致的回归测试）；
（4）延迟交付引发的直接成本；
（5）延迟交付引发的间接成本（例如，机会的损失）。

☆示例：内部失效成本

某软件产品计划卖给 2000 个用户，那么需要为 2000 个用户准备该软件产品的安装软件、安装指南、软件使用指南等，并将这些资料收集刻录到光盘中。那么，软件开发的组织会预约时间，申请 2000 张光盘的制作。假如到预约时间时，研发团队无

法将软件产品相关的资料交付给光盘制作者，组织可能需要支付全部或者部分光盘制作者的等待时间的费用。光盘制作者需要根据组织的新的资料交付时间重新计划和安排光盘的制作。

2.7.4　外部失效成本

外部失效成本指的是在软件产品交付客户之后产生的失效而导致的成本，例如，针对客户的服务成本、针对已经发布的软件版本的补丁开发和在客户中失去良好的声誉等导致的成本。外部失效影响的不仅是软件产品的开发者，同时使用软件产品的客户也要付出由于质量问题而导致的成本。假如软件产品的易用性很差，那么软件产品的使用者需要花费更多的时间去理解和操作软件产品，甚至会在使用过程中出现错误的操作，这些都会给用户带来时间和金钱的损失。和外部失效成本相关的主要活动包括：

（1）技术支持；

（2）保修；

（3）产品召回；

（4）客户调查。

上面介绍了质量成本的 4 个类别：预防成本、检测成本、内部失效成本和外部失效成本。总的质量成本是这 4 个类别的成本的总和。需要注意的是，质量成本的这 4 种类型并不是完全独立的，有时候它们之间是相互重叠的，例如，有些活动既包含预防成本，同时也包含检测成本。

作为质量成本的一个例子，图 2-18 显示了某软件项目 4 个质量成本类别的比例。以图表的格式表达质量成本的构成，读者不仅易于阅读，而且能够很快掌握项目的不同质量成本类别的比例。注意，图 2-18 没有包括质量成本的具体数值，我们无法看出产品或组织的质量成本的变化趋势。

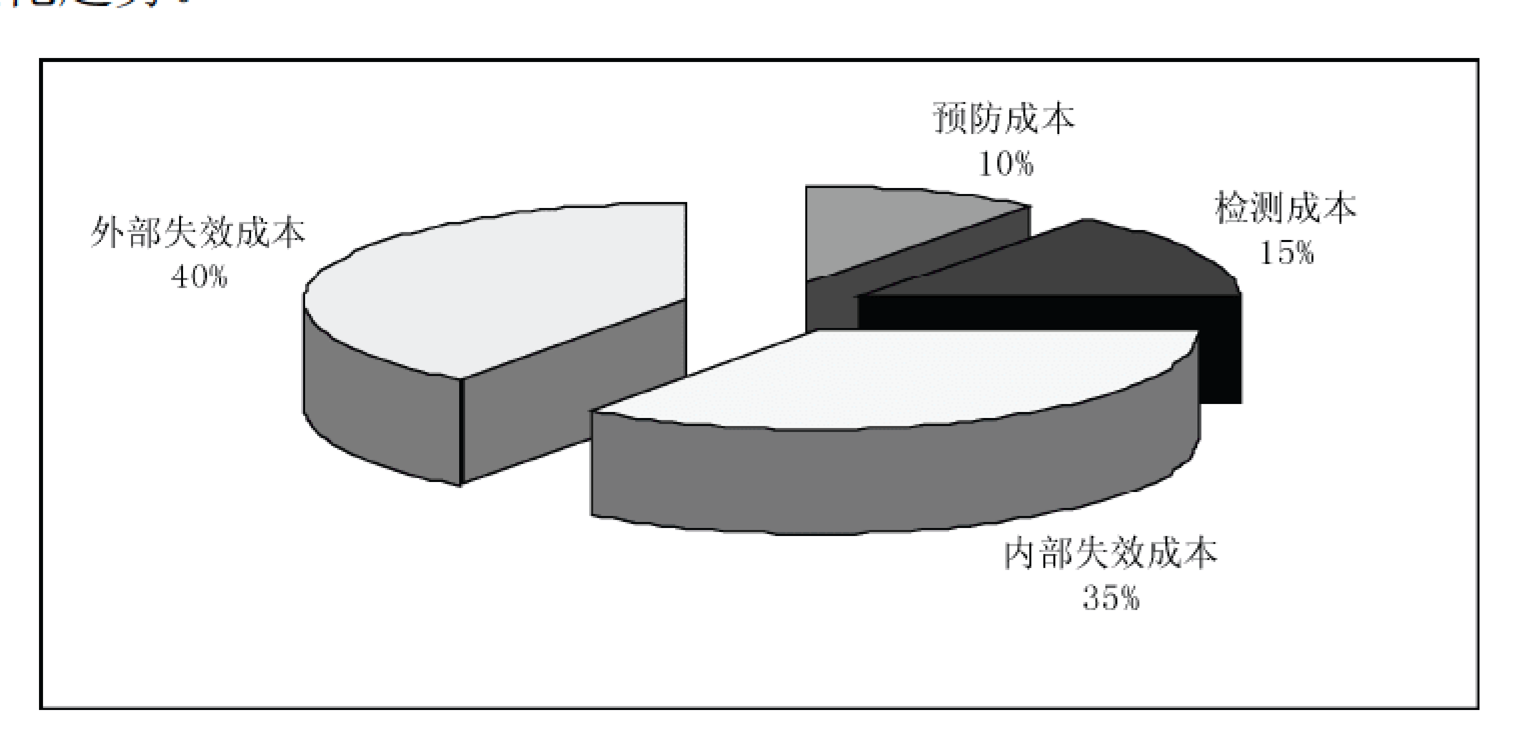


图 2-18　质量成本分布

表 2-19 展示了一个详细的质量成本的总结报告[1]。该报告详细说明了每个质量成本类别对应的成本。这个质量成本的报告在和高级管理人员进行沟通的时候非常有用。从该报

① Steven M．Bragg,Cost Accounting: Comprehensive Guideby,John Wiley & Sons,2001，第 28 章 Cost of Quality。

告可以看出，项目中最大的内部失效成本是材料耗损和返工，而外部失效成本中现场服务和处理客户投诉的成本占了最大的比例。根据需要还可以提供更加详细的质量成本类型，例如，可以将质量成本按照部门或者工作地点进行进一步的细分。

表 2-19　质量成本报告

成本分类	明细	合计
预防成本		
质量管理成本	$7500	
质量培训成本	$2500	
供应商资质审核成本	$4000	
设备预防维护成本	$21 000	
工具设计成本	$2400	
		$ 37 400
检测成本		
接受审查成本	$5900	
测试设备校准成本	$2300	
测试外包成本	$5000	
审查人力成本	$15 000	
测试设备折旧成本	$8100	
		$ 36 300
内部失效成本		
返工成本	$51 000	
材料损耗成本	$43 000	
重新采购的成本	$1900	
检修停工成本	$3500	
供应商不同意见处理成本	$900	
		$100 300
外部失效成本		
产品责任保险成本	$8000	
产品责任成本	$88 000	
保修成本	$29 500	
现场服务成本	$60 200	
处理客户投诉成本	$43 000	
		$228 700
总的质量成本		**$402 700**

质量成本在整个软件项目成本中的比例不是一成不变的。通常来说，随着组织能力的不断提高，质量成本在整个开发成本中的比例会逐渐下降，同时，4 个质量成本类别的比例也会发生一定的变化。图 2-19 是 Knox 创建的一个理论上的质量成本模型[①]，该模型反映了随着组织对应的 CMM 等级的变化，整个质量成本的变化情况。

① Onur Demirors, Ozkan Yildiz, A. Selguk Giicegliogiu, Using Cost of Software Quality for a Process Improvement Initiative, IEEE, 2000。

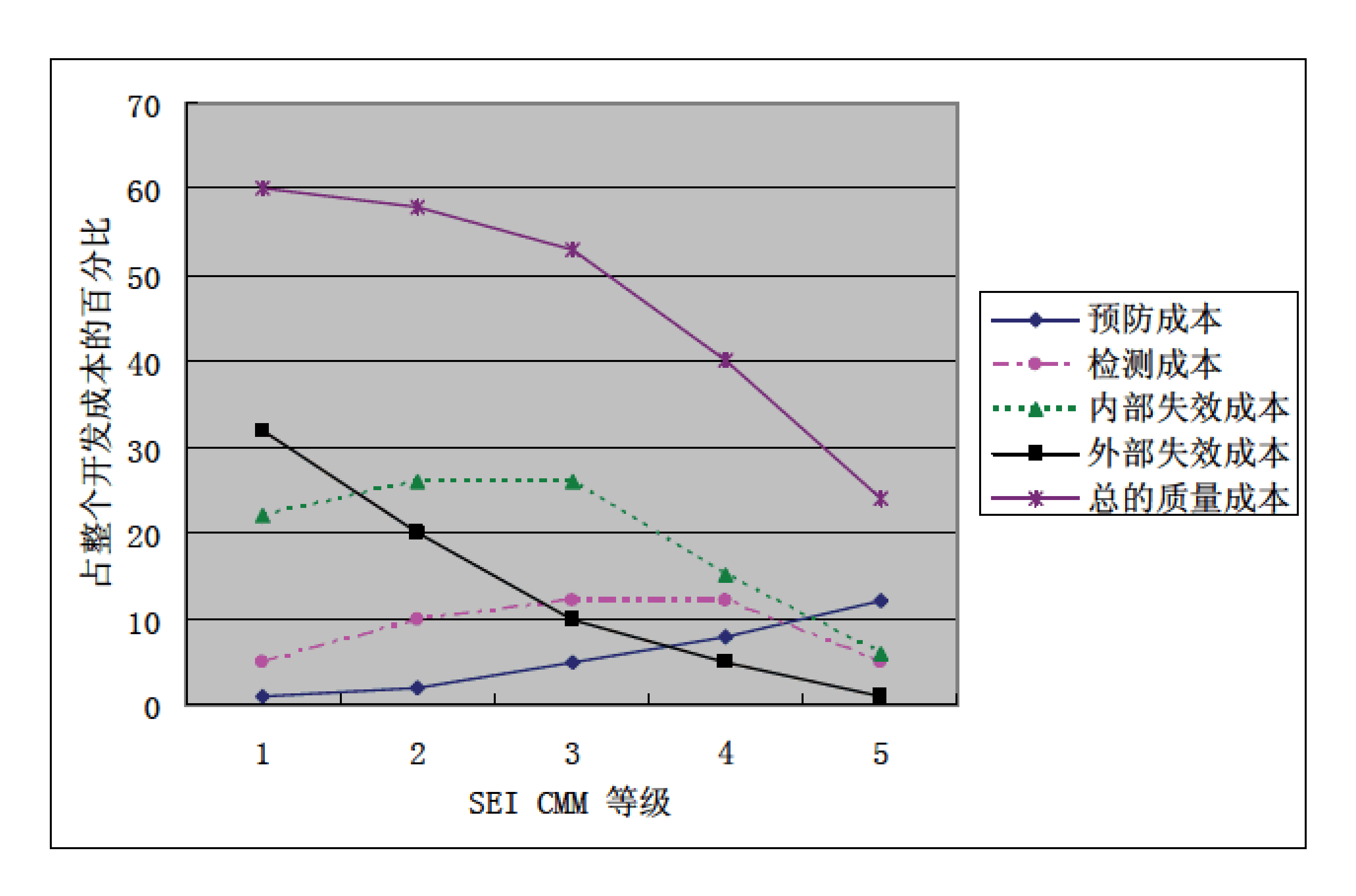


图 2-19 软件质量成本模型

Knox 的模型显示随着组织的 CMM 等级从 1 提高到 5，质量成本占整个开发成本的比例从 60%下降到 25%。同时，随着 CMM 等级的提高，预防成本的比例在不断地上升，而外部失效成本的比例在持续地下降。

☆示例：应用质量成本分析测试的投资回报

下面通过一个假定的场景讲解如何使用质量成本定量分析测试活动的投资回报[①]。假设市场上有个软件产品，每个季度发布一个新版本，平均每个版本包括 1000 个"必须修复"的缺陷。现在假设在开发阶段开发人员发现和修复了 250 个缺陷，而客户发现了剩下的 750 个缺陷。通过分析内部失效成本和外部失效成本，开发人员在开发阶段发现和修复一个缺陷的成本是 10 美元；而在客户使用软件产品过程中发现一个缺陷，开发人员修复它的成本是 1000 美元。如表 2-20 所示，在"没有正式的测试"一列，质量成本是 750 000 美元，它主要是用来处理遗留到客户处的 750 个缺陷。

假设测试人员发现一个缺陷，开发人员修复它需要 100 美元，这个成本是修复客户处发现缺陷的成本的十分之一。假设测试团队在每个软件版本投入的成本是 70 000 美元，在版本发布之前发现了 350 个缺陷。通过计算发现，测试团队投入的增加使得质量成本大幅下降。详细的质量成本数据如表 2-20 所示。

表 2-20 应用质量成本分析测试的投资回报

项目	没有测试的成本	开展测试的成本
测试资源		
人	0	60 000
基础设施	0	10 000
工具	0	0
总投入	0	70 000
开发活动		
发现的必须修复的缺陷数目	250	250
修复成本（内部失效成本）	2500	2500

① 这个例子来自于 Rex Black 的 Investing in Software Testing:The Cost of Software Quality（2000）。

续表

项目	没有测试的成本	开展测试的成本
测试活动		
发现的必须修复的缺陷数目	0	350
修复成本（内部失效成本）	0	35 000
客户支持		
发现的必须修复的缺陷数目	750	400
修复成本（外部失效成本之一）	750 000	400 000
质量成本		
测试投入	0	70 000
非测试投入	752 500	437 500
总的质量成本	752 500	507 500
投资回报率	**NA**	**350%**

测试的商业价值除了可以降低质量成本（包括前面涉及的预防成本、检测成本、内部失效成本和外部失效成本）以外，还表现在测试可以为软件产品发布提供产品状态信息等方面。同时，测试还能够帮助提高软件产品质量的声誉，使软件产品发布更顺利和更可预测，并且可以帮助客户增强和建立对软件产品的信心。

测试能够为管理人员提供产品质量的信息，例如，软件产品是否能够按时发布。随着测试活动的不断进行，收集得到的越来越多的信息可以为管理人员的决策提供帮助。通过对测试广度和深度、覆盖率和发现的缺陷信息的分析，管理人员能够做出更加正确的决定。好的测试有助于了解产品的质量，但是坏的测试给项目带来的只能是一系列的未知和风险。

对于高质量的软件系统，好的测试能够提供足够的信息证明系统的高质量；对于低质量的软件系统，好的测试也能够揭示出该系统的质量很差。但是如果测试做的不好，系统质量的高低就无法进行正确判断，它们的演化关系如图 2-20 所示。

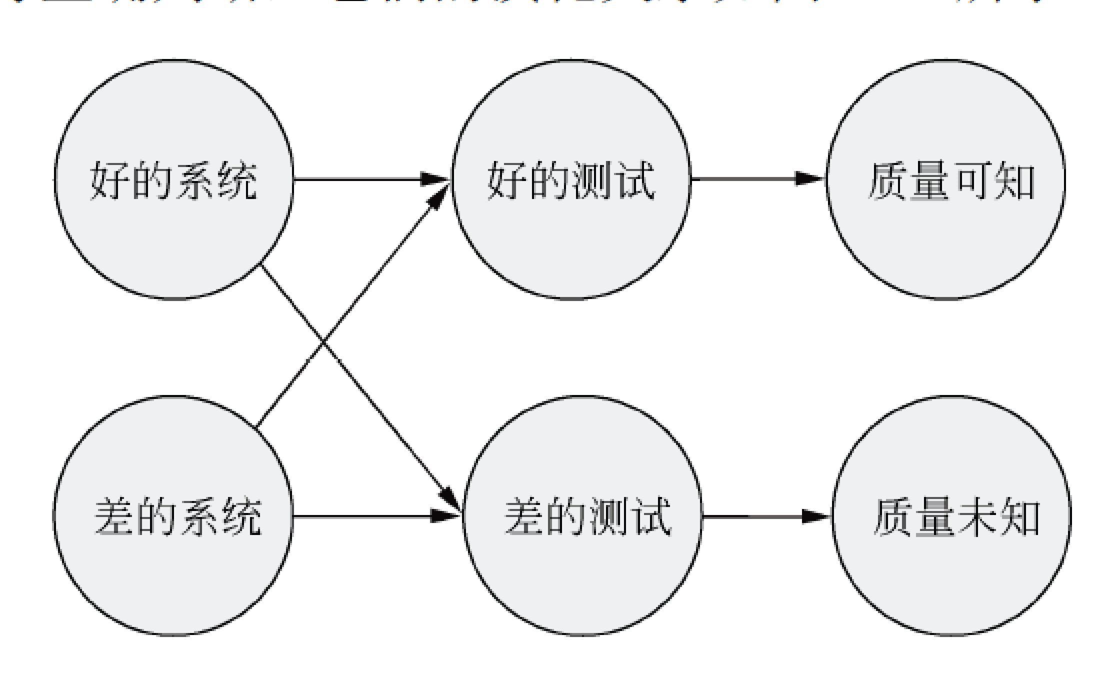


图 2-20　测试和质量的演化关系

软件开发过程中很多重要的决策都要建立在测试提供的信息的基础之上。测试报告中通常都包括测试的范围、测试的深度和测试的结果，在必要的时候，测试的这些信息还可以用来证明软件产品开发满足了合同的要求。虽然软件产品在测试之后仍然会遗留一些缺

陷，但是测试提供的这些信息能够证明整个开发活动符合相关的规章制度，这也是测试的定性商业价值之一。

测试还能够为过程改进提供必要的信息。但是从过程改进的角度看，通过测试获得的信息对于分析组织过程改进的价值很难定量地衡量。通过测试获得的信息可以用来识别组织中需要改进的过程和领域，这些过程既可以是测试过程本身，也可能是其他过程，例如，开发过程。通过过程改进可以提高组织的开发能力，组织中各个项目的遗留缺陷数目会降低，从而提高产品的质量和组织的声誉，有利于组织的长远发展。

更多关于测试过程改进的内容，参见第 5 章。

2.8 分布式测试、外包以及内包测试

正如托马斯·弗里曼（Thomas L.Friedman）所写的，现在的世界是平的。这意味着在今天这样因信息技术而紧密方便的互联世界中，全球市场、劳动力和产品都可以被整个世界共享，一切都有可能以最有效率和最低成本的方式实现。全球化无可阻挡。21 世纪初，很多分析师都犯了一个错误，就是把网络狂飙和全球化混为一谈，以为两者都只是一时的流行。网络泡沫破碎之后，很多分析师都以为全球化也跟着结束。然而事实完全相反，网络狂飙只是全球化的一个层面，泡沫破掉，全球化不但没跟着破，反而更往前冲。幸存的美国科技公司以及那些还想投资初创企业的创投业者都没有太多现金可用。又需要印度工程师了，不是因为他们人多，而是因为他们太便宜。于是美印两国的产业关系就更密切了。

2001 年 12 月 11 日，中国正式加入了 WTO，这意味着中国同意遵守有关进口、出口和外资的全球规范；这意味着中国原则上同意要把自己的竞技场铲得和世界其他地方一样平。自从中国加入 WTO，中国和其他国家都必须开始愈跑愈快。越来越多的跨国公司在中国设立研发中心，中国也在承担越来越多的外包业务[①]。软件测试在全球化的浪潮中也在不断地与时俱进。现在，如果说一个产品的测试任务由在美国和在中国的两个测试团队共同完成，应该不会有人觉得吃惊。

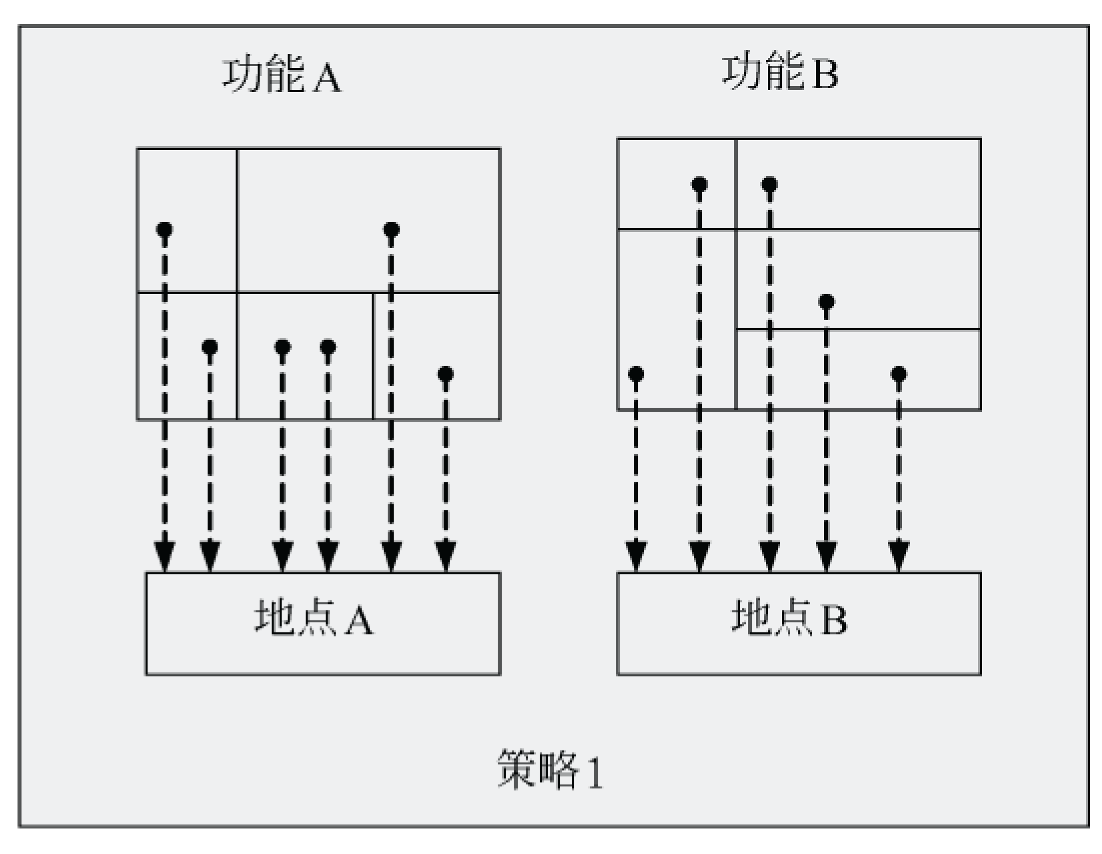
2.8.1 分布式测试

随着全球化程度的不断加强，测试工作很多时候并不是由位于单个地点的测试团队单独完成的，而是由位于多个不同地方的测试团队协作完成的。如果某个软件产品的测试工作由分散在不同地点的测试团队共同完成，称之为分布式测试。

分布式测试意味着整个测试活动在多个不同的地方进行。图 2-21 展示了两种分布式测试的策略。一个项目的测试对象由两大功能组成：功能 A 和功能 B。这两大功能可以分成很多更小的子功能。第一种策略是根据功能划分，不同功能的测试分布在不同的地方，同一个地方的测试团队负责相同的功能；第二种策略是把两个不同的测试团队当成一个整体分派测试任务，而不考虑他们地点的差距，同一个功能的不同子功能可能分别在不同的地方进行测试。第一种策略的优点是管理比较简单，但其缺点是资源利用可能无法最大化；

① (美)托马斯·弗里德曼．世界是平的．何帆，肖莹莹，译．长沙：湖南科学技术出版社，2006.

第二种策略的优点是可以较好地利用测试资源，但其缺点是测试团队之间的高效沟通是个大的挑战，因此如果没有丰富的跨区域的管理经验和良好的沟通协调技巧，很可能导致项目的失败。



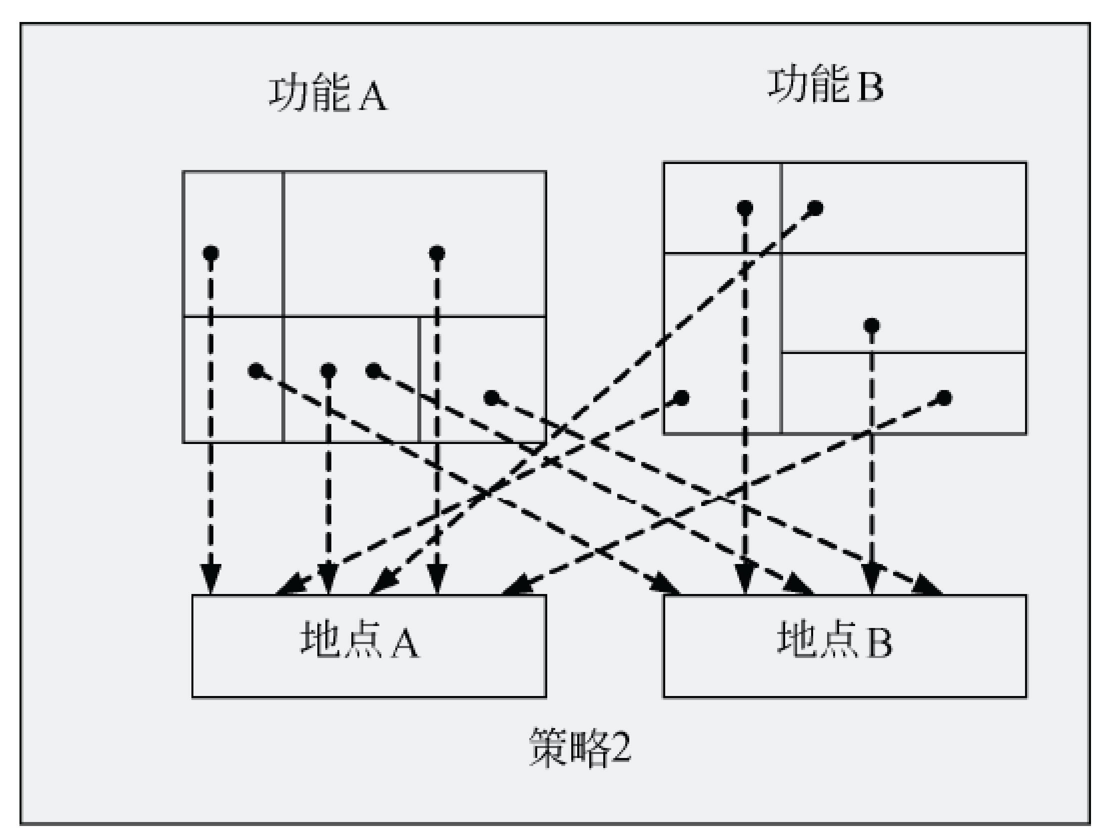


图 2-21　分布式测试

分布式测试中位于不同地点的测试团队可以是同一个组织的不同分部，也可以是外包的其他测试团队。

2.8.2　外包测试

若测试工作由其他（有别于项目团队所在地）一个（或多个）地点的非本组织成员承担，称之为外包测试，更准确地说，应该是离岸外包，如图 2-22 所示。

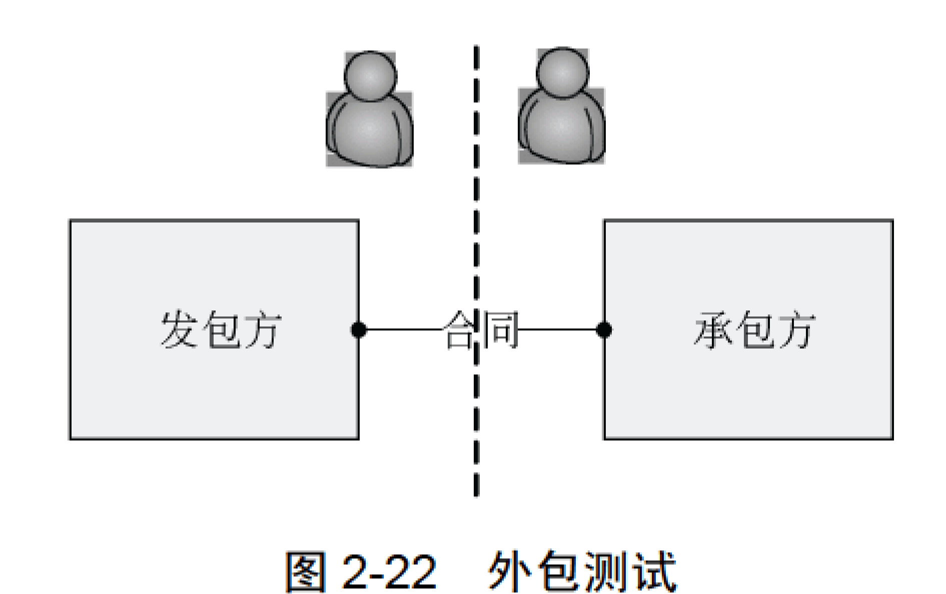


图 2-22　外包测试

外包测试中测试任务由另外一个或多个承包方的测试团队完成，且测试地点和发包方不在同一个地方。外包测试在全球范围内被广泛使用。外包测试具有以下优点。

（1）降低成本：很多采用外包测试的组织的初衷都是为了降低成本。通过外包测试可以大大降低发包方的成本开支，这些开支主要包括劳动力开支、招聘费用、国家保险费用和工作场所等。仅劳动力开支这一项就可以降低很多。发达国家的员工劳动力成本很高，而在发展中国家，随着其竞争力的不断加强，同样素质的劳动力需要的成本要比发达国家低很多。

（2）保证质量：随着电信业迈向先进的数据化服务，技术熟练而薪酬要求较低的员工对企业非常重要。在印度等发展中国家，受过良好教育的员工甚至比美国本土的员工具有更高的素质，这些员工的加盟对企业的产品和服务质量的提升大有益处；同时，承包方长期从事软件测试，对软件测试有很深的认识，能够提供更高的质量，降低测试的风险。

（3）提高效率：工作团队遍布世界不同的时区，可以全日 24 小时不停地运作，缩短开发应用方案所需的时间。外包测试可以更好地利用软件和硬件资源，由于时区的不同，通过合理地选择外包团队的时区，可以大大提高工作的效率。例如，产品开发团队在美国硅谷，而测试团队外包给中国上海的某个公司，那么美国的开发人员在下班前交付一个版本，上海的测试团队在对该版本验证后，可以在美国开发人员第二天上班前提交测试结果，非常有利于加快项目的进度。

（4）灵活的人力资源：承包人员从组织上并不属于发包方，所以发包方不需要对承包人员的长期发展直接负责，通常，双方的合同到项目结束就会中止。这样发包方不会因为没有后续项目而产生人员闲置的情况。尤其是在当今世界经济剧烈动荡的情况下，外包测试更有利于降低人力资源方面的风险。

2.8.3 内包测试

若测试工作由其他与项目团队在同一地点的非本组织成员承担，称之为内包测试，如图 2-23 所示。

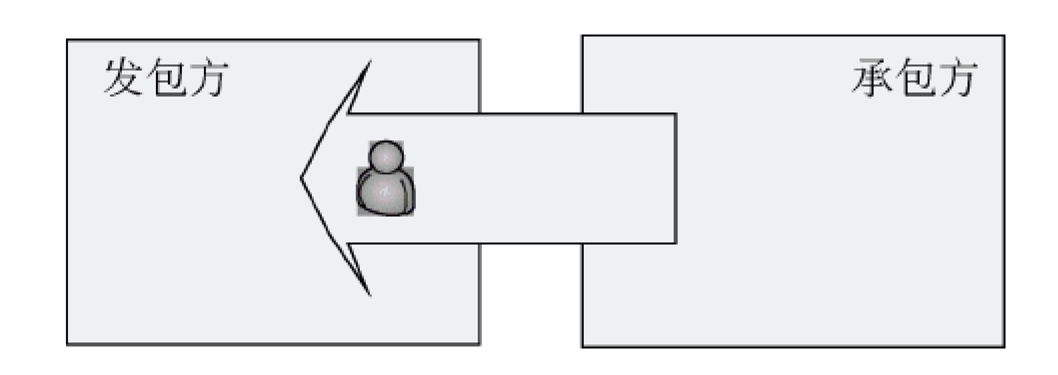


图 2-23 内包测试

内包测试和外包测试不同的是，承包方的工作人员和发包方在相同的工作地点，承包方把工作人员直接派遣到发包方的工作场所。内包测试这种方式，发包方必然会有成本的增加，例如，发包方要为承包方的工作人员提供额外的工作场所和相关办公环境，以及可能需要负担额外的生活成本，如果发包方和承包方处在不同的国家，还要考虑签证的通过率和有效期等问题。内包测试除了具有外包测试的保证质量、提高效率和灵活的人力资源以外，还有一些外包测试不具有的优点。

（1）发包方可以采用相同的管理方式和过程，很方便对承包人员进行管理。

（2）能够快速建立工作环境，并及时对承包人员进行相关的培训。

（3）可以从承包人员处学习到很多原来不具备的专业技能。

（4）测试内包经常在长期外包测试之前采用，这样不仅有利于提高承包人员的技能，还可以为将来的长期外包测试奠定基础。

2.8.4 风险

虽然分布式测试、外包测试和内包测试都有各自的优点，但是这并不意味着可以随意采用这些不同的测试方式。它们在成本核算、外包地点选择、承包商选择、测试管理和商业模式方面都面临着新的挑战。在采用这些不同的测试方式之前，组织内部要进行详细的评估，判断采用不同测试方式的性价比。分布式测试、外包测试和内包测试通常会面临如下 5 个方面的风险[①]。

1．沟通

测试团队的选择和管理需要考虑沟通方面的风险。在测试管理过程中，需要明确不同团队之间的交流方式，并明确定义各个团队的使命、任务和交付内容。项目团队必须减少对非正式交流渠道的依赖，如在走廊上的对话或同事之间的私人交流。地点、时区、文化和语言的差异使得有效地沟通变得更加重要。当测试团队和其他团队之间存在时区差距的时候，及时沟通方面可能存在一定的问题。例如，当位于美国的项目人员已经下班，而中国的测试人员正在进行测试工作，一旦在测试活动中碰到任何问题，都没有办法得到美国项目相关人员的及时支持。这种情况下可能需要安排专门的人员值班，尽量保证双方的工作时间有一定的交叉，以方便相关问题和项目进度状态的交流。如果大家的母语不同，那么沟通的效果也会打折扣。在中国，虽然英语已经相当普及，但是仍然有很多人的英语听力和口语还都不能满足工作的需要。当测试团队位于不同国家的时候，语言问题可以通过安排一个接口人来解决，这个接口人需要具备较高的英语水平和沟通能力；而当采用内包测试的时候，语言问题将会更突出，这种情况下通常都要求承包方和发包方的工作语言保持一致，这样对承包方人员的整体语言水平要求更高。

2．监管

对不同团队进行有效监管是一个挑战，监管的缺乏也是一个潜在的风险。由于测试团队和项目团队在不同的地方可能隶属于不同的组织，监管工作显得尤为重要。对监管工作的重要性认识不足，常常导致监管的人力不够，或者监管的人缺乏足够的技能。监管活动主要包括以下几项。

首先要组建监管团队，监管团队成员要具备必要的技能，监管团队中仅包括测试人员是不够的，虽然测试人员在技术方面比较精通，但是在监管过程中还会涉及很多非技术方面的问题，所以必须有团队成员具备相应的管理技巧、商业经验和社交技能等。

其次，测试团队的服务质量可能随着时间的推移发生变化，这就要求监管团队定期对测试团队的工作质量进行评估，为了避免不必要的纠纷，最好是在开始签订合同的时候就

① Johann Rost 在 *The Insider's Guide to Outsourcing Risks and Rewards*（Auerbach Publications，2006）一书中对外包的风险进行了详细的分析。

对交付物的质量和测试团队的服务等级进行定义。对于内包测试，由于没有地域的差别，可以比较方便地实时对所有日常的测试活动进行监控；对于外包测试和分布式测试，由于存在地域的差别，可以通过外派监管团队的方式对整个测试活动进行监控，同时还可以通过对测试活动的中间交付物进行检查保证测试的质量，例如，对设计的测试用例进行评审、对编写的测试代码进行抽查、检查测试执行的日志等。

最后，在所有的测试活动结束后，还需要按照合同约定的验收条件对最终产品进行确认，这可能需要执行一系列的验收测试来保证产品的质量。

3．保密

由于涉及不同地点或组织的参与人员，机密信息的保护显得尤为重要。所有项目相关的人员都需要签署保密协议，明确定义所有项目参与者的保密责任和违约的法律责任。对于分布式测试，如果位于不同地点的测试团队只是该项目所在组织的一个研发分支，那么保密工作要稍微简单一点儿。如果涉及不同组织的外部人员参与项目，保密工作就要花费更多的时间和成本。为了保证保密工作的顺利进行，发包方要严格定义相关人员的角色和职责，从而根据不同的职责分配相应的权限。如果采用内包测试的形式，还可以通过为承包方人员提供办公用品（办公场所、网络和计算机等）来减少泄密的可能性。对于外包测试的形式，除了避免承包方项目人员的泄密，还要注意双方数据传输过程中的信息保密。在采用外包测试的时候，不可避免地要进行各种信息的传送，可能是双方的电话、E-mail交流，也可能是软件版本的传输，在条件允许的情况下要尽量使用 VPN 等方式。如果有必要，对传输的数据要进行加密。

4．成本变化

成本是动态变化的，一段时期或单个项目的成本只是发包方应该考虑的因素之一。通常情况下，发包方都希望和承包方建立一种长期合作的关系。如果是同一个组织内的分布式测试，就希望能够长期建立一个稳定的研发分支。如果是采用外包测试或者内包测试，发包方也希望能够和承包方维持长期的合作关系，毕竟重新选择一个合作伙伴会带来额外的成本和风险。

另外，参与外包测试和内包测试的测试人员的薪水在逐步提高，发达国家和发展中国家薪水的差距也在不断缩小。在人力成本不断上升的同时，随着承包方的不断强大，他们也会对自己的定价策略进行调整，承包方认为高质量应该意味着高价格，外包方式的成本优势会逐步减弱。

5．潜在的竞争

对于外包测试和内包测试，发包方还要评估它们的潜在竞争威胁。随着承包方承担工作的增加，发包方对承包方的依赖程度可能会不断增强，这种情况下，承包方完全可能变成一个强势的地位，而出现抬高价格的现象。面对这样的风险，发包方可以在条件允许的情况下同时选择多个承包方，从而尽量避免对单个承包方的过分依赖。同时随着承包方实力的不断增强，他们完全可能成为发包方潜在的竞争对手。这就要求发包方仔细评估外包的项目，通常只将非核心的任务进行外包，同时在外包过程中加强知识产权的保护工作。

综上所述，分布式测试、外包测试和内包测试由于地域的差别和人员组织的不同，给管理工作带来了很大的挑战。如果测试团队之间或测试团队、开发团队、管理团队之间使用各自不同的方法，那么将造成显而易见的问题，尤其是在测试执行期间。对于分布式测

试，不同的工作地点之间的测试工作的划分必须要明确、合理。如果不按这个原则分配，就可能导致各个测试团队分配不到其最适合的测试工作，进而使整个测试工作受到隔阂、间隙（增加产品发布的质量风险）和重叠（降低效率）的困扰。

最后，对于上述涉及的各种方式的测试，整个项目团队对测试团队的信任非常重要。要相信各个测试团队虽然在组织、文化、语言和地理位置上有所不同，但他们依旧能很好地履行他们的职责。团队之间缺少信任会导致效率低下和时间延迟。缺少信任可以表现在包括验证彼此的活动、相互追究问题的责任以及组织内的小团体等。

2.9　管理行业标准的使用

ISTQB 基础级大纲和高级大纲参考了许多标准和规范，涉及软件开发生命周期、软件测试与测试方法论、软件质量特征、评审和缺陷管理等。本高级大纲无意列出或者推荐特定的标准。测试经理应该了解标准是如何创立的，在工作环境当中如何使用，以及使用这些标准可能的收益与风险等。标准可以有不同的来源，例如：

（1）国际标准或具有国际性目标的标准；

（2）国家标准，如国际标准在某个国家的国内版；

（3）专业领域特定的标准，如对国际标准或国家标准进行裁剪，以应用到特定的领域，或是为某个专门的领域开发的国际或国家标准。

2.9.1　标准的来源和有效性

1．标准的来源

标准的制定通常是一个长期和艰难的过程。标准是由专业人士建立的，并反映了集体的智慧。目前常见的标准的来源包括：国际标准、国家标准和特定领域的标准。国际标准有两个主要来源：电气和电子工程师协会（IEEE）和国际标准化组织（ISO）。国家标准有很多，不同的国家通常都有自己的标准，而一些特定的领域也会有一些专门的标准。

☆示例：中华人民共和国标准法的部分描述

对需要在全国范围内统一的技术要求，应当制定国家标准。国家标准由国务院标准化行政主管部门制定。对没有国家标准而又需要在全国某个行业范围内统一的技术要求，可以制定行业标准。行业标准由国务院有关行政主管部门制定，并报国务院标准化行政主管部门备案，在公布国家标准之后，该项行业标准即行废止。对没有国家标准和行业标准而又需要在省、自治区、直辖市范围内统一的工业产品的安全、卫生要求，可以制定地方标准。地方标准由省、自治区、直辖市标准化行政主管部门制定，并报国务院标准化行政主管部门和国务院有关行政主管部门备案，在公布国家标准或者行业标准之后，该项地方标准即行废止。

企业生产的产品没有国家标准和行业标准的，应当制定企业标准，作为组织生产的依据。企业的产品标准须报当地政府标准化行政主管部门和有关行政主管部门备

案。已有国家标准或者行业标准的，国家鼓励企业制定严于国家标准或者行业标准的企业标准，在企业内部适用。

测试人员需要了解在测试工作和环境中所使用的标准是正式标准（国际、国家或者特定领域的标准）还是内部标准、推荐实践方法。标准也在不断演变和修改，在使用或引用标准时需要指出标准的版本号（或者发布日期）以保证其一致性。在标准的参考文献使用中，假如没有指明使用文献的版本号以及发布日期的，须使用最新发布版本。

2．标准的有效性

标准可以作为提高产品质量的工具，用于减少缺陷的引入而非只是在测试当中发现缺陷。并不是所有的标准都适用于所有的项目，标准中的信息可能对项目有益，也可能妨碍项目。标准可以为项目提供好的方法，也可以为项目提供好的过程。在组织内部实施合适的标准有助于提高组织的测试能力，保证产品质量，同时也有利于组织在行业内的认可。

☆示例：中华人民共和国标准法部分内容

第八条 制定标准应当有利于保障安全和人民的身体健康，保护消费者的利益，保护环境。

第九条 制定标准应当有利于合理利用国家资源，推广科学技术成果，提高经济效益，并符合使用要求，有利于产品的通用互换，做到技术上先进，经济上合理。

第十条 制定标准应当做到有关标准的协调配套。

第十一条 制定标准应当有利于促进对外经济技术合作和对外贸易。

标准并不是描述了一个科学的真理，而是由一群专家根据自己的经验和业界的实践制定出来的。这就意味着标准之间可能存在不一致，甚至是冲突。即使是在一些基本概念的定义上，各个标准之间可能也是不一致的。这时标准的使用者应该根据其使用环境和背景决定不同标准的实用性。

2.9.2 国际标准

1．ISO

国际标准化组织（ISO）是目前世界上最大、最有权威性的国际标准化专门机构之一。1946 年 10 月 14 日至 26 日，中、英、美、法、苏等 25 个国家的 64 名代表集会于伦敦，正式表决通过建立国际标准化组织。1947 年 2 月 23 日，ISO 章程得到 15 个国家标准化机构的认可，国际标准化组织宣告正式成立。参加 1946 年 10 月 14 日伦敦会议的 25 个国家为 ISO 的创始人。ISO 是联合国经社理事会的甲级咨询组织和贸发理事会综合级（即最高级）咨询组织。此外，ISO 还与六百多个国际组织保持着协作关系。国际标准化组织的目的和宗旨是：“在全世界范围内促进标准化工作的发展，以便于国际物资交流和服务，并扩大在知识、科学、技术和经济方面的合作。”其主要活动是制定国际标准，协调世界范围的标准化工作，组织各成员国和技术委员会进行情报交流，以及与其他国际组织进行合作，共同研究有关标准化问题①。

① 本节关于 ISO 的介绍来自于国家标准化管理委员会：http://www.sac.gov.cn/templet/default ShowArticle.jsp?id=2622。更详细的内容也可以参考 ISO 的官方网站 www.iso.org。

国际标准化组织是由各国标准化团体（ISO 成员团体）组成的世界性的联合会，也为软件测试人员提供了许多有用的标准，例如：

（1）ISO/IEC 9126:1998，现已分成以下的标准和技术报告（TR）。

① ISO/IEC 9126-1:2001 软件工程—产品质量—第 1 部分：质量模型。

② ISO/IEC TR 9126-2:2003 软件工程—产品质量—第 2 部分：外部度量。

③ ISO/IEC TR 9126-2:2003 软件工程—产品质量—第 3 部分：内部度量。

④ ISO/IEC TR 9126-2:2004 软件工程—产品质量—第 4 部分：质量度量。

（2）ISO 12207:1995/Amd 2:2004 系统和软件工程—软件生命周期过程。

（3）ISO/IEC 15504-2:2003 信息技术—过程评定—第 2 部分：执行评定。

以上所列并没有涵盖 ISO 所有可供使用的软件测试相关的标准，测试人员可根据其工作内容和环境选择其他 ISO 标准。

2. IEEE

电气和电子工程师协会（IEEE）是一个国际性的电子技术与信息科学工程师的协会，拥有来自 160 个国家的 37.5 万会员（截至 2008 年 12 月 31 日），1963 年 1 月 1 日，由美国无线电工程师协会（IRE，创立于 1912 年）和美国电气工程师协会（AIEE, 创建于 1884 年）合并而成，它是一个区域和技术互为补充的组织结构，以地理位置或者技术中心作为组织单位（例如：IEEE 费城分会和 IEEE 计算机协会），总部在美国纽约市。IEEE 拥有 329 个地方分会。透过多元化的会员，该组织在航空航天、计算机、电信、生物医学、电力及消费性电子产品等领域中都是主要的权威之一。IEEE 发表多种杂志、学报、书籍，并在每年组织三百多次专业会议。IEEE 定义的标准在工业界有极大的影响力。学会成立的目的在于为电气电子方面的科学家、工程师、制造商提供国际联络交流的场合，方便他们交流信息，并提供专业教育和提高专业能力的服务。学会的主要活动是召开会议、出版期刊杂志、制定标准、继续教育、颁发奖项、认证等[①]。

IEEE 为测试人员提供了许多测试相关的有用标准，例如：

（1）IEEE 610:1991 IEEE 标准计算机词典，汇集了 IEEE 标准计算机词汇；

（2）IEEE 829:2008 IEEE 软件测试文档标准；

（3）IEEE 1028:2008 IEEE 软件评审标准；

（4）IEEE 1044:1993 IEEE 软件异常分类；

以上所列并没有涵盖 IEEE 所有可供使用的软件测试相关的标准，测试人员可根据其工作内容和环境选择其他的 IEEE 标准。

2.9.3　国家标准

许多国家本身制定了符合自身特点的特定标准，其中的某些标准可用于软件测试，例如，英国的 BS-7925-2：软件测试和软件组件测试（Software testing, Software component testing）提供了组件测试的过程描述以及许多关于测试技术的信息，具体包括：

（1）等价类划分；

① 更多关于 IEEE 的详细介绍，参见 IEEE 的官方网站：http://www.ieee.org/web/aboutus/home/index.html。

（2）边界值分析；
（3）状态转换测试；
（4）因果图；
（5）语句测试；
（6）分支/判定测试；
（7）条件测试；
（8）随机测试。

☆示例：中国标准分类

A 综合	B 农业、林业
C 医药、卫生、劳动保护	D 矿业
E 石油	F 能源、核技术
G 化工	H 冶金
J 机械	K 电工
L 电子元器件与信息技术	M 通信、广播
N 仪器、仪表	P 土木、建筑
Q 建材	R 公路、水路运输
S 铁路	T 车辆
U 船舶	V 航空、航天
W 纺织	X 食品
Y 轻工、文化与生活用品	Z 环境保护

上面的列表是中国国家标准的分类。其中和软件测试相关的标准重点集中在：L 电子元器件与信息技术。该类下面还有很多子类。

L00/09 电子元器件与信息技术综合	L10/34 电子元件
L35/39 电真空器件	L40/49 半导体分立器件
L50/54 光电子器件	L55/59 微电路
L60/69 计算机	L70/84 信息处理技术
L85/89 电子测量与仪器	L90/94 电子设备与专用材料、零件、结构件
L95/99 电子工业生产设备	

L77 是和软件工程相关的标准类。下面的这三个标准都属于 L77 软件工程类的国家标准。

（1）GB/T 8566—2007（信息技术 软件生存周期过程）：该标准为软件生存周期过程建立了公共框架，以供软件产业界使用。包括在含有软件的系统、独立软件产品和软件服务的获取期间以及在软件产品的供应、开发、运行和维护期间需应用的过程、活动和任务。

（2）GB/T 9386—2008（计算机软件测试文档编制规范）：该标准规定了一组基本的计算机软件测试文档的格式和内容要求。本标准适用于计算机软件生存周期的全过程。

（3）GB/T 15532—2008（计算机软件测试规范）：该标准规定了计算机软件生存周期内各类软件产品的基本测试方法、过程和准则，适用于计算机软件生存周期的全过程。

当然，和软件测试相关的中国国家标准还有很多，详细的信息可以查阅中国国家标准化管理委员会的官方网站（http://www.sac.gov.cn/templet/default/）。

2.9.4　特定领域标准

不同的技术领域当中同样存在着不同的标准。一些行业根据其特定的技术领域背景，对其他标准进行了裁减，并形成新的标准。本节提及的内容仍然集中在软件测试、软件质量和软件开发等方面。

1. 航空电子系统

RTCA DO-178B/ED 12B：航空系统和设备认证软件的注意事项（Software Considerations in Airborne Systems and Equipment Certification）适用于民用飞机中所使用的软件。1982 年，RTCA 和 EUROCAE 正式发布了 DO-178，这是民用航空机载软件开发中安全保证的一个里程碑。这个标准有两个称谓，在美国（RTCA）被称为 DO-178，在欧洲（EUROCAE）被称为 ED-12。其实，这完全是同一个标准。在接下来的几年里，依据 DO-178 进行开发和认证的经验表明，DO-178 还不太完善，需要修订。1985 年，新的版本 DO-178A 就问世了，与之等价的欧洲版本为 ED-12A。

DO-178A 有一个很大的特点，它面向的是软件的开发技术和开发方法。但是，软件的开发技术更新很快，新的技术和方法层出不穷，日新月异。DO-178A 很快就显得跟不上步伐了。为了解决这个问题，RTCA 决定再次修订这个标准。为了避免重蹈覆辙，RTCA 和 EUROCAE 的专家们改变了制定这个标准的原则，从原来的“面向开发技术和方法”改成“面向目标”和“面向进程”。这样，他们决定完全重写这个标准。终于，一个相对稳定的版本：DO-178B 在 1992 年问世了，至今还在应用中。①

现在，DO-178B 是国际公认的民用航空机载软件的开发标准。一架民用飞机（相对军用飞机而言）不经过“民航标准体系”的适航认证，是不可以飞行的。而这个“民航标准体系”中，针对机载软件适航认证的就是 DO-178B 标准，由此可见 DO-178B 的重要性。

DO-178B 把软件开发分成 5 个主要的过程，分别是：软件计划、软件开发、软件验证、软件配置管理和软件质量保证。每一个过程都有严格的文档输出的要求。DO-178B 强调了过程数据的重要性。整个软件生命周期中所有过程的活动和结果都必须被记录下来。

☆示例：DO-178B 中的软件生命周期数据

DO-178B 对软件生命周期的数据有明确的要求。软件生命周期数据包括各个过程的输出数据，其中有一项就是和软件测试用例相关的，它要求每个测试用例都应该包括输入、执行步骤、期望结果以及成功/失败的准则，同时还要求编写详细的测试规程来描述每个测试用例是如何进行准备并被执行的，以及如何对测试结果进行评估。

2. 航天工业

有一些工业领域会对其他的标准进行裁减以适合他们特定的领域，例如，航天工业中的 ECSS（欧洲空间标准化合作组织，www.ecss.nl）。根据软件的关键等级，ECSS 推荐了

① 更多关于 DO-178 的内容，可以参考 http://www.do-178b.cn/viewthread.php?tid=2&extra=page%3D1。

一系列的方法和技术，包括：

（1）SFMECA——软件失效模式、影响以及危急程度分析；

（2）SFTA——系统失效树分析；

（3）HSIA——软硬件相互分析；

（4）SCCFA——软件常见失效分析。

3．食品与药物管理局

美国食品和药物管理局（Food and Drug Administration，FDA）是美国政府在健康与人类服务部（DHHS）和公共卫生部（PHS）中设立的执行机构之一。在国际上，FDA 被公认为世界上最大的食品与药物管理机构之一。其他许多国家都通过寻求和接受 FDA 的帮助来促进并监控其本国产品的安全。

FDA 于 2002 年发布了“软件确认的基本原则”。该标准对医疗设备软件的验证活动进行了规范。具体包括软件生命周期、需求分析、缺陷预防、变更管理和独立评审等多个方面的内容。

2.9.5 其他标准

除了上面提及的国际标准、国家标准和特定领域标准之外，还有许多适用于不同产业领域的标准，其中有些是修改后用于特定的产业或者领域，也有一些适用于特定任务或者为如何使用标准提供解释。

测试人员要了解适合自己所处的专业领域、产业和工作环境的不同标准（包括内部标准、最佳实践等）。根据特定合约，各个标准的实用性有所不同。测试经理最终决定所需要遵循的标准，确保和维护标准的充分执行。

小结

测试管理贯穿于整个测试过程，是测试经理的主要职责之一。测试经理应当清楚了解测试过程中的各个测试阶段、活动和任务，并对它们进行有效的管理。本章主要从 9 个方面阐述了测试管理的内涵。

“一定条件下的测试管理”章节主要描述了测试相关的主要项目干系人、其他软件开发生命周期活动及工作产品、测试活动与其他生命周期是如何集成的、如何有效管理非功能性测试，以及如何管理基于经验的测试等内容，以及它们是如何影响测试活动和测试效率的。

“基于风险的测试和其他测试优先级设定和工作量分配的方法”章节主要描述了基于风险的测试、基于风险的测试技术、测试优先级设定和工作量分配的其他技术，以及测试过程中的测试优先级设定和工作量分配等实践活动。

“测试文档和其他工作产品” 章节详细描述了与 IEEE Std 829—2008 定义兼容的测试方针、测试策略、主测试计划和级别测试计划，以及探讨了如何在测试过程中有效管理项目风险，其他与测试活动相关的工作产品等内容。并根据组织结构、产品风险、项目风险、

产品规模和类型等因素，裁减 IEEE Std 829—2008 中建议的测试计划内容。

“测试估算”章节描述了测试估算的基本概念以及影响测试估算的各个因素。通过具体的项目案例详细地分析各种可行的测试估算技术，例如，基于团队的测试估算、基于百分比的测试估算、基于测试规模的测试估算等。

“定义和使用测试度量”章节描述了测试过程监控的 5 个维度：产品风险、测试、覆盖率、缺陷和信心。并讲解如何利用在监控活动中观察到的测试过程相关的信息和问题，制定应对计划控制当前的测试过程，并提供改进建议。

“测试的商业价值”章节描述了质量成本的基本含义，讲解了预防成本、检测成本、内部失效成本和外部失效成本的基本概念。并从定量分析和定性分析两个方面分析具体的质量成本。

“分布式测试、外包测试和内包测试”章节简单介绍了分布式测试、外包测试和内包测试的基本概念，并对这三种测试类型各自的优缺点进行了分析。

“管理行业标准的使用”章节主要阐述了与测试活动紧密相关的国际标准、国内标准和特定行业标准，以及各类型标准主要的侧重点。

模拟题

10．CTAL-ATM_LO-2.2.1

TM-2.2.1(K4) 分析软件项目或程序的干系人、环境、需求，包括软件开发生命周期模型，并识别最佳测试活动。

问题：

场景：

假设你在管理一个成熟应用软件的测试。该应用软件提供在线约会服务，可以允许用户输入他们自己的简介，以找到与他们匹配的人；安排用户与合适对象的社交活动；并且屏蔽他们不愿意联系的人。

考虑下面不同类型的利益干系人：

Ⅰ．使用该应用软件寻找约会对象的用户

Ⅱ．公司内部的管理人员和项目干系人

Ⅲ．使用该应用软件找到了配偶的已婚夫妻

Ⅳ．政府部门的员工

考虑下面的测试活动：

a．测试该应用软件建议的匹配对象之间的相互吸引程度

b．测试该应用软件针对用户收费正确性的能力

c．测试该应用软件是否遵守当地的税收法规

仅根据上面提供的信息，下面哪个选项的利益干系人和他们的测试关注点最匹配？

选项：

A．Ⅰ-a, b; Ⅱ-a, b, c; Ⅳ-c

B．Ⅰ-a, b; Ⅱ-a, b, c; Ⅲ-b; Ⅳ-c

C．Ⅰ-a, b; Ⅱ-a, b, c; Ⅳ-a, c

D．Ⅰ-a, b, c; Ⅱ-a, b, c; Ⅳ-c

解释：

A．正确。用户关心的是他们的付费（以他们同意的价格），是否得到了应得的服务；管理层和项目干系人必须关心所有这三类的测试，这样他们才能让客户满意，让公司盈利并遵守法律法规；政府部门关心是否遵守法律法规；已婚的夫妻不是当前的利益干系人。

B．不正确。已婚的客户不是当前的客户（除非他们隐瞒他们的婚姻状态），因此他们不会真正关心收费的正确性。

C．不正确。政府员工并不真的关心对象匹配功能是否可以很好工作，除非那些员工也是该应用的用户（这与是否是属于政府部门的员工没有关系）。

D．不正确。用户不太关心该公司是否已合理交税，他们只关心是否被正确扣费。

分值：3 分

11．CTAL-ATM_LO-2.2.2

TM-2.2.2(K2) 理解软件开发生命周期中的活动和工作产品如何影响测试，以及测试如何影响软件开发生命周期中的活动和工作产品。

问题：

下面哪个选项正确反映了项目管理过程的工作产品如何影响测试活动？

选项：

A．项目计划中的资源限制，可能会影响测试活动

B．测试经理应该与项目经理一起制定项目进度

C．测试应该完全覆盖需求规格说明

D．在测试结束阶段测试经理应该与技术支持经理一起工作

解释：

A．正确。测试计划必须与更大的项目计划保持一致。

B．不正确。选项 B 本身是正确的论述，但是它描述的是测试如何影响项目管理工作产品，而不是项目管理工作产品影响测试。

C．不正确。需求规格说明不是项目管理工作产品，另外，该论述只有在采用基于需求的测试策略情况下才正确。

D．不正确。选项 D 本身是正确的论述，但它不是关于项目管理影响测试的论述，而是测试如何影响技术支持。

分值：1 分

12．CTAL-ATM_LO-2.2.3

TM-2.2.3(K2) 解释在基于经验的测试和非功能性测试中处理测试管理问题的方法。

问题：

下面哪个选项描述了管理非功能测试的合适方法？

选项：

A. 如果非功能测试实现活动需要超过一整个迭代的时间，则应该在迭代之外进行
B. 测试经理应该将非功能测试计划授权给该项目的技术测试分析师（TTA）
C. 非功能测试的优先级应该按照功能测试和识别的风险进行排列
D. 非功能风险应该在早期的测试级别甚至是开发阶段采取缓解活动

解释：

A. 正确。来自大纲。
B. 不正确。只有一部分测试计划可以委派给 TA 和 TTA。
C. 不正确。所有的非功能测试不一定需要遵循功能测试的顺序（但应该基于识别的风险）。
D. 不正确。有些非功能风险可能在早期缓解，但有些需要在软件开发生命周期的后期缓解。

分值：1 分

13. CTAL-ATM_LO-2.3.1

TM-2.3.1（K2）描述基于风险的测试响应风险的各种不同方式。

问题：

下面哪个选项最好地描述了基于风险的测试是如何根据风险做出响应的？

选项：

A. 测试团队设计、实现和执行测试用例以缓解质量风险
B. 通过测试发现缺陷，提高了被测系统的质量
C. 功能测试主要应对产品风险，而非功能测试应对质量风险
D. 测试经理基于项目风险确定采用的测试级别

解释：

A. 正确。参见大纲。
B. 不正确。测试可以测量质量，但测试不能直接改善质量（后续的调试将会改善质量）。
C. 不正确。风险类型与功能/非功能测试之间没有什么关系（尽管它们属于相同风险类型）；

D．不正确。项目风险可以帮助确定采用什么测试级别，产品风险同样也可以。

分值：1 分

14．CTAL-ATM_LO-2.3.2

TM-2.3.2(K2) 举例说明产品风险分析的不同技术。

问题：

下面哪个选项是分析产品质量风险的不同技术的例子？

选项：

A．PRAM, PRiSMa, FMEA 和 FTA

B．风险识别，风险评估，风险缓解和风险管理

C．专家评审，独立评估，使用风险模板和项目回顾会议

D．业务分析师，设计师和程序员之间存在的个人和培训问题

解释：

A．正确。属于大纲中提供的 8 种不同技术中的 4 种。

B．不正确。这些属于基于风险的测试的 4 个活动。

C．不正确。这些属于风险识别的技术，只是分析过程的一部分。

D．不正确。这是影响风险项可能性的一个因素。

分值：1 分

15．CTAL-ATM_LO-2.3.3

TM-2.3.3(K4) 分析、识别和评估产品质量风险，从关键项目干系人的角度总结风险及其评估的风险级别。

问题：

假设你为某个刚启动的项目工作。该项目构建一个可以提供定制忠诚度与奖励计划的系统，小企业和中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。允许公司创建定制化的按钮，放在他们的网站，从而允许客户将他们的信息登记到公司的忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，确定获得免费产品或者服务所需的积分。

公司的销售人员对该系统进行了大幅度的促销，针对最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明该服务，针对公司和客户，都可以提供高可靠性和极快的响应速度。

在目前阶段，需求已经完成，软件开发工作刚刚开始。当前的进度计划将允许公司和他们的客户在三个月之后开始登记。

公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员和其他的工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

下面哪三个选项属于该软件产品的产品质量风险？

选项：

A．由于内存泄漏，系统性能直线下降
B．忠诚度点数计算不正确
C．客户不能在公司忠诚度计划页面注册
D．在测试开始之前，该创业公司的资金就用光了
E．系统的需求描述不正确
F．云计算资源不能及时用于支持项目计划
G．第一年的大幅度促销导致公司现金流危机

解释：

A．正确。我们承诺提供高可靠性，这属于质量特性。
B．正确。计算忠诚度点数是系统功能之一，而功能准确性是一个质量子特性。
C．正确。公司忠诚度计划页面注册属于系统功能，而功能准确性是一个质量子特性。
D．不正确。属于项目风险（针对任何创业公司都非常现实的风险）。
E．不正确。属于项目风险，需要学员能够分辨两者之间的细微区别：风险的潜在影响（对项目造成的各种间接损害）和风险作为输出而导致的各种后果（包括各种质量相关的问题）。
F．不正确。属于项目风险，不是质量风险。同时由于云计算租赁市场存在各种可选项，该风险的可能性非常小。
G．不正确。绝对是风险，但和产品的质量没有关系，而只是由于提供的折扣造成的。这属于版本发布之后出现的运行风险。

分值：3 分

16．CTAL-ATM_LO-2.3.4

TM-2.3.4(K2) 描述在软件开发生命周期和测试过程中，怎样根据评估的风险级别，适当地缓解和管理识别的产品质量风险。

问题：

针对如何对识别的产品质量风险进行缓解和管理的描述，下面哪个描述是**最不合适**的？

选项：

A．选择需要遵循的监管标准会受到风险级别的影响
B．为了解决预期的风险，应该设计、实现和执行测试
C．测试开发和执行的工作量应该与预期的风险级别成比例
D．测试开发和执行的优先级应该基于预期的风险级别

解释：

A．正确。我们通常要求测试的高有效性（即检查缺陷的能力），不管缓解的是什么级别的风险。
B．不正确。参见大纲。
C．不正确。参见大纲。
D．不正确。参见大纲。
分值：1 分

17．CTAL-ATM_LO-2.3.5

TM-2.3.5(K2) 举例说明测试选择、优先级设定和分工的不同方案。

问题：

下面哪个**不是**针对测试优先级和工作量分配的实践技术？

选项：

A．单个测试人员根据在测试依据中发现的缺陷决定要测试什么
B．模糊性评审用于识别和消除需求中的模糊问题
C．测试条件分析包含认真阅读各优先级的需求，以识别需要覆盖的测试条件
D．采用因果图技术识别达到测试依据 100%功能覆盖的测试集

解释：

A．正确。不要将所有的优先级和工作量分配依赖于单个测试人员，而且发现缺陷只能发生在测试开始之后（优先级和工作量分配同样如此）。
B．不正确。参考大纲内容。
C．不正确。参考大纲内容。
D．不正确。参考大纲内容。
分值：1 分

18．CTAL-ATM_LO-2.4.1

TM-2.4.1(K4) 分析给出的测试方针和测试策略，建立主测试计划、级别测试计划和与这些文档相补充和相一致的其他测试工作产品。

问题：

场景：

假设你在管理一个成熟应用软件的测试。该应用软件提供在线约会服务，可以允许用户输入他们自己的简介，以找到与他们匹配的人；安排用户与合适对象的社交活动；并且屏蔽他们不愿意联系的人。

假设测试组织的测试方针中，按照优先级定义了如下任务。

（1）发现缺陷；

（2）降低风险；

（3）建立信心。

假设你的经理已经定义了明年测试组织在测试过程改进中最高的优先级是最大程度的对应用软件的回归测试实现自动化。

下面哪个描述是正确的？

选项：

A．应用软件与测试过程改进相一致，但是任务描述与该应用软件和测试过程改进不一致

B．应用软件与任务描述相一致，但是测试过程改进与应用软件和任务描述都不一致

C．应用软件、任务描述与过程改进都一致

D．应用软件、任务描述与过程改进两两都不一致

解释：

A．正确。对于成熟的应用软件，测试的主要任务是真正地建立信心，即该应用软件可以持续正常工作。自动化回归测试有助于提高效率，因此测试过程改进与应用软件相一致。尽管针对该成熟应用的自动化回归测试是一个很好的想法，但是自动化无法找到很多的缺陷。因此，测试过程改进和任务描述不一致，或者说和成熟应用软件的实际测试要求不一致。

B．不正确。原因参考选项 A 的描述。

C．不正确。原因参考选项 A 的描述。

D．不正确。原因参考选项 A 的描述。

分值：2 分

19．CTAL-ATM_LO-2.4.2

TM-2.2.4(K4) 针对给定的项目，分析项目风险并选择适当的风险管理方案（如缓解、应急、转移和/或接受）。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

假设你正在编写该项目的主测试计划，当前正在考虑计划的项目风险章节。

下面哪个主题**不应该**包含在测试计划的项目风险章节？

选项：

A．在推广该应用服务之前，高级营销人员辞职

B．在计划的测试执行开始日期之前，无法配置测试环境

C．无法提供足够的有技能和经过认证的测试人员，特别是高级测试人员

D．没有足够的资源获得适合数量的虚拟用户进行压力测试

解释：

A．正确。这是一个严重的项目风险，但不是与测试相关的项目风险。营销团队已经完成了测试团队所需的需求。

B．不正确。测试环境是否准备完毕的问题，属于典型的测试相关的项目风险。

C．不正确。测试人员合格和可用问题，属于典型的测试相关的

项目风险。

D．不正确。工具是否准备完毕，属于典型的测试相关的项目风险。

分值：2 分

20．CTAL-ATM_LO-2.4.3

TM-2.4.3(K2) 描述并举例说明测试策略如何影响测试活动。

问题：

考虑下面的测试策略：

Ⅰ 分析式的测试策略

Ⅱ 基于方法的测试策略

Ⅲ 符合过程或标准的测试策略

Ⅳ 咨询式的测试策略

考虑下面的测试活动：

1 测试用户提供的 Internet 浏览器列表

2 定义用户故事的验收准则

3 尽早执行最高风险的测试

4 单击网页上所有的导航链接

下面哪个选项正确地将测试策略与测试活动关联在一起？

选项：

A． Ⅰ-3； Ⅱ-4； Ⅲ-2； Ⅳ-1

B． Ⅰ-1； Ⅱ-2； Ⅲ-3； Ⅳ-4

C． Ⅰ-4； Ⅱ-3； Ⅲ-2； Ⅳ-1

D． Ⅰ-3； Ⅱ-2； Ⅲ-4； Ⅳ-1

解释：

A．正确。因为分析式的测试策略包括基于风险的测试，基于风险级别安排测试优先级；基于方法的测试使用检查表（在该例子中包括遍历网站的各个链接）；符合过程的测试可以包括敏捷过程的一致性；咨询式的测试包括依赖于外部指导的方式。

B．不正确。因为这些例子中至少有两个不匹配。

C．不正确。因为这些例子中至少有两个不匹配。

D．不正确。因为这些例子中至少有两个不匹配。

分值：1 分

21．CTAL-ATM_LO-2.4.4

TM-2.4.4(K3) 制定适合组织、生命周期以及项目需要的测试工作产品的文档规范和模板，适时裁剪标准的主体部分，生成可用模板。

问题：

场景：

假如你是项目测试经理，该项目采用敏捷开发生命周期。测试策略混合了基于风险的测试、符合过程的测试和应对型测试。开发人员遵循已知的敏捷最佳实践，包括自动化组件测试和持续集成。

你正在定义编写各种测试工作产品的指南。

下面哪个论述是正确的？

选项：

A．你根据各种资源（包括 IEEE Std 829 标准）裁剪得到一系列的模板

B．你应该遵循 IEEE Std 829 标准，因为你遵循过程符合的测试策略

C．你应该遵循 IEEE Std 829 标准，因为它可用于任何行业

D．你可以忽略所有测试工作产品的文档化，除了缺陷报告

解释：

A．正确。敏捷开发生命周期强调轻量级的文档化。

B．不正确。这里的符合过程要求指的是敏捷方法，而不是 IEEE Std 829 标准。

C．不正确。IEEE Std 829 标准属于重量级文档化，在文档方面与敏捷思想以及应对型测试策略是不符合的。

D．不正确。即使是应对型的测试，也会有测试章程，即使敏捷生命周期也会有验收准则。

分值：2 分

22．CTAL-ATM_LO-2.5.1

TM-2.5.1(K3) 针对给定的项目，使用所有适当的估算技术创建整个测试过程活动的估算。

问题：

场景：

假如你是项目测试经理，该项目采用敏捷开发生命周期。测试策略混合了基于风险的测试、过程符合的测试和应对型测试。开发人员遵循已知的敏捷最佳实践，包括组件测试自动化和持续集成。

你正在估算测试团队所负责的特定迭代的系统测试工作量。

下面哪三个选项正确地描述了你应该如何针对上述场景进行估算？

选项：

A. 根据以前迭代，考虑每个识别的风险平均所需的工作量
B. 针对每个识别的测试章程，分配固定时间的测试
C. 针对**该迭代，参加估算相关的敏捷会议**
D. 只针对测试执行和测试结束活动估算工作量
E. 估计在系统测试执行阶段，能发现绝大多数的缺陷
F. 包括创建详细测试工作产品文档所需的工作量
G. 假设系统测试可以重用组件测试的数据与环境

解释：

A. 正确。估算过程中考虑历史平均数据属于认可的估算技术。
B. 正确。这是常见的管理基于经验的测试的技术，可以指导估算工作。
C. 正确。测试人员应该作为敏捷团队的一员参与到估算活动中。
D. 不正确。测试估算应该包括测试过程的所有活动。
E. 不正确。根据大纲的内容，开发人员遵循已知的敏捷最佳实践，在系统测试之前能移除至少一半的缺陷。
F. 不正确。敏捷方法避免采用过于详细的文档，包括测试文档。
G. 不正确。场景中没有涉及任何关于重用方面的要求。
分值：3 分

23. CTAL-ATM_LO-2.5.2

TM-2.5.2(K2) 理解可能影响测试估算的因素，并举例说明。

问题：

下面哪个因素最可能影响测试需要的时间，但是不会影响测试活动的工作量？

选项：

A. 修复测试阶段发现的缺陷所需的时间
B. 测试过程的成熟度
C. 测试条件所需的详细程度
D. 所需的系统质量

解释：

A. 正确。缺陷修复所需的时间可能会导致测试延期，但是不会影响测试的工作量。
B. 不正确。即使是成熟的测试过程，假如工作量大，持续的测试时间就会比较长。

C．不正确。测试条件细化需要工作量。
D．不正确。质量要求的系统，需要更多的测试工作量。
分值：1 分

24．CTAL-ATM_LO-2.6.1

TM-2.6.1(K2) 描述并比较典型的测试相关度量。

问题：

下面哪个关于测试度量使用的描述是正确的？

选项：

A．再测试和回归测试的状态可以用来监控测试进度
B．使用从报告缺陷到解决缺陷的滞后时间趋势来奖惩开发人员
C．识别的测试条件数目可以用来监控测试质量
D．开发测试件的计划时间和实际时间可以用来最小化回归测试

解释：

A．正确。这属于测试进度监控度量的一个。两者都属于测试度量，但是用于不同的对象。
B．不正确。参考使用该度量的危害性。
C．不正确。属于监控测试分析进度的度量。
D．不正确。属于监控计划和控制的度量。
分值：1 分

25．CTAL-ATM_LO-2.6.2

TM-2.6.2(K2) 比较不同层面的测试进度监控。

问题：

下面哪个选项最适合用来监控测试进度？

选项：

A．覆盖率、信心、风险、测试和缺陷度量的组合使用
B．通过测量已经执行的测试数目估算代码覆盖率
C．覆盖率、产品、人员、测试和缺陷度量的组合使用
D．产品、人员和项目度量的组合使用

解释：

A．正确。提到了测试进度度量维度中的 5 个。
B．不正确。将错误的覆盖率对象关联到已经执行的测试数目上。

C. 不正确。包括人员度量的类别，它们通常不用来监视测试进度；另外，人员和产品度量没有被列在测试进度度量的 5 个维度之中。

D. 不正确。包括人员度量的类别，它们通常不用来监视测试进度。

分值：1 分

26. CTAL-ATM_LO-2.6.3

TM-2.6.3(K4) 分析和汇报测试结果，主要包括剩余风险、缺陷状态、测试执行状态、测试覆盖状态及信心以提供见解和建议，帮助项目干系人做出发布决策。

问题：

你在国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线采用增量产品开发生命周期，一个长期的传统是构建紧凑的集成产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年时间，大部分人具有更长的工作时间。经验丰富的测试人员利用公司内部的定制测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时还对所有业务单元的个人效率进行度量。你的公司在公司级别实施了 CMMi。

团队在赶上硬件开发进度方面存在问题。软件业务团队的经理要求你提供建议改进项目的测试，例如引入更好的度量或者工具。管理层很快从用户代表那里收集了产品风险列表，他们认为测试没有覆盖所有的风险。

下面哪个选项应该是你建议完成的？

选项：

A. 基于测试人员的信心分析剩余风险，以检查是否达到了足够的覆盖率

B. 增加更多的测试以更好地覆盖功能

C. 根据测试人员对开发人员开发能力的评价来获取风险和信心状态

D．将信心等级加入度量

解释：

A．正确。它综合使用了更多不同的测试进度监控，并开始寻找可能影响测试覆盖率与所做工作导致剩余产品风险的真正原因。

B．不正确。B 选项本身是正确的，但是没有进行分析，不能仅依赖于直觉进行工作。

C．不正确。错误地关注在能力方面。

D．内容本身是正确的，但是没有使用风险的信息。

分值：2 分

27．CTAL-ATM_LO-2.7.1

TM-2.7.1(K2) 分别给出决定质量成本 4 种类别的例子。

问题：

考虑下面的质量成本分类：

a 预防成本

b 检测成本

c 内部失效成本

d 外部失效成本

考虑下面的质量成本例子：

1 开展质量风险分析

2 为业务分析师培训软件需求工程

3 客户抱怨性能太差

4 测试阶段从发现缺陷到修改缺陷的周期太长，导致缺陷管理效率低下

下面哪个选项，正确地匹配了质量成本分类与例子？

选项：

A．a-2; b-1; c-4; d-3

B．a-1; b-2; c-3; d-4

C．a-1; b-2; c-4; d-3

D．a-2; b-1; c-3; d-4

解释：

A．正确。培训业务分析师是预防缺陷，它可以帮助编写高质量的需求；产品质量风险分析属于检测成本，即使没有发现缺陷也需要花费成本；在测试阶段和发布之前的所有与缺陷相

关而导致的成本，即使是可避免的成本都属于内部失效成本；客户抱怨属于外部失效成本，因为客户抱怨会导致降低将来的销量。

B. 不正确。参考选项 A 的解释。

C. 不正确。参考选项 A 的解释。

D. 不正确。参考选项 A 的解释。

分值：1 分

28. CTAL-ATM_LO-2.7.2

TM-2.7.2(K3) 基于质量成本，以及其他定性和定量的考虑，估算测试的价值，并将其告知测试干系人。

问题：

场景：

假设你在管理一个成熟应用软件的测试。该应用软件提供在线约会服务，可以允许用户输入他们自己的简介，以找到与他们匹配的人；安排用户与合适对象的社交活动；并且屏蔽他们不愿意联系的人。

假设你计算得到了下面的质量成本。

（1）平均检测成本：$150。

（2）平均内部失效成本：$250。

（3）平均外部失效成本：$5000。

平均检测成本和内部失效成本是通过在发布之前发现的缺陷数目计算得到的；而外部失效成本是通过在发布之后发现的缺陷数目计算得到的。

下面哪个描述是正确的？

选项：

A. 测试阶段发现的每个缺陷，都可以为组织平均节省质量成本 $4600

B. 针对该交际应用软件的总质量成本，包括预防成本，为$5400

C. 无法利用质量成本对该组织或任何其他组织，进行测试价值的计算

D. 测试阶段发现的每个缺陷，都可以为组织平均节省质量成本 $5400

解释：

A. 正确。$5000－($150＋$250)＝$4600。

B. 不正确。无法将平均质量成本相加得到一个总数，因为没有得到预防成本的数据；

C. 不正确。质量成本可以用来计算任何质量相关活动的价值，这在**世界**范围的行业内都在使用。

D. 不正确。必须减去与测试相关的平均检测成本和内部失效成本，从而计算潜在的净节约，而不是将它们相加。

分值：2 分

29. CTAL-ATM_LO-2.8.1

TM-2.8.1(K2) 理解成功运用分布式团队、外包团队和内包测试团队策略的所需因素。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

为了通过限制测试团队的员工数量以降低测试成本，高层管理者决定引入外包测试服务公司负责部分测试工作。实际的测试工作在马来西亚完成，该公司派遣一位员工常驻以方便直接协调工作，沟通测试结果，以及与离岸测试团队进行每天两次的联系。

针对分布式和外包测试，在该计划中提到了下面哪个成功因素？

选项：

A. 定义沟通的方式

B. 根据资质分配测试工作
C. 为现场和离岸的测试团队定义良好的使命和任务
D. 在项目团队成员之间建立信任关系

解释：

A. 正确。现场的一个员工作为沟通的渠道。
B. 不正确。除了成本，这里没有提到任何其他因素用来选择外包测试服务公司。
C. 不正确。这里没有在测试团队之间进行明确的工作划分。
D. 不正确。采用离岸团队是自上而下强行决定的，因此从最开始的时候就伤害了信任关系。

分值：1 分

30. CTAL-ATM_LO-2.9.1

TM 2.9.1(K2) 总结软件测试标准的来源及使用。

问题：

下面哪个选项最好地描述了提供了实现测试覆盖率指南的软件标准？

选项：

A. 美国联邦航空管理局的标准 DO-178B
B. ISTQB 大纲和术语
C. CMMi 软件过程改进框架
D. PRINCE2 项目管理框架

解释：

A. 正确。参考大纲内容。
B. 不正确。不是标准，没有提供测试覆盖率标准的指南。
C. 不正确。通用的过程改进模型，没有提供测试覆盖率标准的指南。
D. 不正确。通用的项目管理指南，没有提供测试覆盖率标准的指南。

分值：1 分



第3章 评审

本章学习目标如表 3-1 所示。

表 3-1 学习目标

编 号	学习目标描述	级 别
TM-3.2.1	了解管理评审和审计的关键特性	K2
TM-3.3.1	分析项目，选择适宜的评审类型并定义开展评审的计划，确保评审得到适当的实施、跟进，并确定责任	K4
TM-3.3.2	了解参与评审需要的因素、技能和时间	K2
TM-3.4.1	定义评审使用的过程和产品度量	K3
TM-3.5.1	举例说明正式评审的特点	K2

本章相关术语如表 3-2 所示。

表 3-2 相关术语

英 文	中 文	说 明
audit	审计	对软件产品或过程进行的独立评审，以确认产品是否满足标准、指南、规格说明书以及基于客观准则的步骤等，包括下面的文档：① 产品的内容与形式；② 产品开发应该遵循的流程；③ 度量符合标准或指南的准则[IEEE 1028]
informal review	非正式评审	一种不基于正式（文档化）过程的评审
inspection	审查	一种同级评审，通过检查文档以发现缺陷，例如不符合开发标准，不符合更上层的文档等。审查最正式的评审技术，是基于文档化的过程[IEEE 610, IEEE 1028]。参见 peer review[GBT 11457]
management review	管理评审	由管理层或其代表执行的对软件采购、供应、开发、运作或维护过程的系统化评估，包括监控过程、判断计划和进度表的状态、确定需求及其系统资源分配，或评估管理方式的效用，以达到正常运作的目的 [IEEE 610, IEEE 1028]
moderator	主持人	负责审查或其他评审过程的负责人或主要人员
review	评审	对产品或产品状态进行的评估，以确定与计划的结果所存在的误差，并提供改进建议。例如，管理评审、非正式评审、技术评审、审查和走查与（IEEE 1028 一致）[GBT 11457]
review plan	评审计划	项目评审计划，它描述了评审方法、资源和进度安排，除此之外，明确了评审的文档、代码、评审类型、参与者、进入/退出标准，同时对计划制定的依据进行了说明
reviewer	评审员	参与评审的人员，辨识并描述被评审产品或项目中的异常。在评审过程中，可以选择评审人员从不同角度评审或担当不同角色
technical review	技术评审	一种同行间的小组讨论活动，主要为了对所采用的技术实现方法达成共识[Gilb and Graham, IEEE 1028]。参见 peer review
walkthrough	走查	由文档作者逐步陈述文档内容，以收集信息并对内容达成共识[Freedman and Weinberg, IEEE 1028]。参见 peer review[GBT 11457]

3.1 简介

评审属于静态测试的范畴，测试经理需要对其成功实施负责，尤其是针对软件开发生命周期中输出的工作产品时。组织测试方针需要在更广泛的软件项目层面，明确定义评审的过程、角色和职责。正式评审可以在软件开发生命周期的任何阶段实施，而且可以应用于不同的领域和软件产品。评审的负责人可以是测试经理、质量保证经理、有资质的测试人员，也可以是经过培训的评审协调员。在本书中，评审的负责人一律称为评审主持人。

评审的成功与否会受到各种因素的影响，因此评审主持人应该确保当前的环境条件与ISTQB 基础级大纲中定义的确保评审成功实施的因素相符合。评审主持人需要编制评审度量计划，从而对评审进行评估并确保评审达到既定目标。

具备特定软件产品业务技能和技术技能的利益干系人参与评审是评审成功实施的重要条件。而测试人员通常同时具备针对测试对象的业务技能和技术技能，例如，了解用户使用软件产品的操作行为和理解该软件产品所必须具备的功能特性等，因此他们积极参与评审过程也是非常重要的。

为了更加高效地开展评审活动，评审参与人员应该接受正规的评审培训，包括评审过程（评审过程的 6 大阶段、每个阶段的主要活动等）、评审参与的角色与职责、不同评审类型的特点与目标等。所有评审的参与者都必须尽力完成各自需负责的职责要求，从而为评审带来良好的效益。

软件开发生命周期中每个阶段输出的不同软件工作产品，都可以作为评审对象，并对它们实施不同的评审类型。假如评审过程成功实施，评审是一种非常有效的质量保证手段，不仅可以提高软件产品质量和降低成本，还可以加快软件项目的进度。而评审主持人对评审过程的监控和协调，以及证明评审的投资收益率，对成功实施评审尤为重要。评审主持人根据开发过程中不同阶段输出的工作产品的特点和质量要求，可以开展不同的评审类型，例如：

（1）合同评审，软件项目开始启动时和项目主要里程碑点可以开展；

（2）需求评审，需求文档初稿且满足一定条件下即可发起评审，通常需要同时覆盖功能和非功能需求；

（3）概要设计评审，软件产品整体架构设计满足评审条件时即可发起评审；

（4）详细设计评审，详细设计规格说明满足评审条件时即可发起评审，通常是在编码工作开始之前发起；

（5）代码评审，软件模块生成代码时进行，可以包括单元测试结果以及代码本身；

（6）测试工作产品评审，可以覆盖测试计划、测试条件、质量风险分析结果、测试用例、测试数据、测试环境和测试结果等；

（7）每个测试级别的测试入口准则评审和测试出口准则评审，分别在测试执行开始前检查测试入口准则，结束测试执行前检查测试出口准则；

（8）验收评审，用于获取客户或利益干系人对软件产品是否满足要求的批准。

不同类型的测试技术，其侧重点是不一样的，例如，评审可以在编码之前发现需求中

存在的问题，因此可以避免将该问题引入到后续的代码中；而静态分析可以协助编码规范的检查，并发现通过人工检查工作产品难以识别的一些问题；而审查不仅可以发现和移除缺陷，还能训练作者如何避免在其工作产品中引入缺陷。

不同的评审类型可以结合使用，例如，可以先对测试用例规格说明进行走查，后续再进行技术评审。评审的主要目的是尽早发现其中的缺陷，但评审是实现该目的的手段之一。假如评审结合其他静态测试手段，例如，静态分析，或者代码的动态测试，可以更好地发挥不同技术在发现缺陷方面的优点，并弥补各自的不足，从而可以提高测试覆盖率，并识别更多的缺陷。

ISTQB 基础级大纲把评审作为静态测试技术引入，包括：非正式评审、走查、技术评审和审查，其对象是软件开发生命周期中的各种工作产品。而测试经理还经常需要参与管理评审和审计，它们也属于静态测试范畴。相对于前面的 4 种评审类型，管理评审和审计更关注过程，而非软件工作产品。

3.2 管理评审和审计

正如前面所述，管理评审和审计也属于静态测试的范畴，相对软件开发生命周期中输出的工作产品，其更关注过程。那么，管理评审和审计有什么不同特点呢？

3.2.1 管理评审

管理评审的目的是监控项目过程、确定项目计划和执行进度状态，或者评估为获取某个目的而采用的管理方法的有效性。管理评审可以帮助确定合适的纠正措施、项目资源分配的变更，或者项目范围的变更。

管理评审可以用来识别当前项目状态和计划的一致性或者是否存在偏差，采取的管理方法是否合适以及是否满足要求等。在管理评审过程中，相关的技术知识可以帮助管理评审达到目的。管理评审的对象可以是软件工作产品（例如，测试执行状态），也可以是软件开发相关的过程（例如：缺陷管理过程）。

管理评审的主要参与人有经理、主持人、记录员、作者、评审员等，在需要的时候，也可以包括其他项目利益干系人，例如，其他小组成员、客户代表等。管理评审的过程满足通用的评审 6 大阶段，分别是计划管理评审、管理评审的预备会、评审员的个人准备、管理评审会议召开、返工阶段和跟踪结果阶段。

管理评审的输出应当文档化，其中应该包括管理评审的对象（工作产品还是过程）、参与管理评审的成员名单、管理评审的目标、管理评审的相关输入、管理评审得到的应对活动列表（应对活动的状态、负责人、完成的目标时间、完成的实际时间等）以及管理评审识别的缺陷列表等。

管理评审的关键特征如下。

（1）目的：监控进度、评估状态、为下一步行动做出决策。

（2）由对软件项目或系统负有直接责任的经理执行。

（3）由项目利益干系人或决策者执行，例如，更高级别的经理或总监。

（4）核对与计划的一致性、与计划之间的偏离度，或者管理规程的充分性。

（5）评估项目风险。

（6）评估应对行动的影响和衡量影响的方法。

（7）结果包括行动应对计划、待解决的问题列表和决策列表。

（8）参与者要求进行各项准备工作，必须在文档中记录相关的决策。

测试经理不仅是管理评审的积极参与者，有时候也是管理评审的发起人，例如，针对测试进度的管理评审。另外，软件开发生命周期中实施的管理评审，例如，项目回顾评审（经验教训会议），通常是过程改进的组成部分。

3.2.2 审计

审计是非常正式的，其目的是提供独立的评估，证明工作产品和过程与规范、标准、指南、计划、规格说明和规程等的一致性。因此，相对其他类型的评审，审计在揭示缺陷方面是效率最低的。

审计的主要参与人员包括审计负责人、记录员、审计员、审计发起者和被审计的组织。审计角色中的审计发起者确定审计的具体需求。审计发起者选择审计组织开展独立的审计评估活动。同时审计发起者提供给审计员相关的信息，包括审计的目的、被审计的工作产品或者过程、评估标准等。审计负责人生成审计计划，而审计员准备审计活动。

审计的主要特征如下。

（1）目的：对过程的一致性、规范性、标准等提供独立的评估。

（2）审计负责人对审计活动负责，并且担任审计主持人的角色。

（3）审计员通过谈话、仔细检查文档，收集是否满足一致性要求的证据。

（4）审计结果包含观察资料、建议、纠正活动和通过/不通过的结论。

3.2.3 案例分析：成功实施评审活动

评审可以提高软件工作产品的质量，提高发现和修复缺陷的有效性，并为后续的开发活动、测试活动和管理控制活动提供更好的可预测性。在评审过程中，评审成员之间的讨论和解释可以作为一种有效的培训手段，而评审过程中收集的数据和信息可以为将来软件项目的缺陷预防提供重要的依据。因此，评审应该作为重要的静态测试技术引进组织中。为了成功地将评审引入到项目和组织中，需要采用以下的一些措施。

（1）获得管理层支持：评审需要时间和资源的承诺，例如，评审员的时间计划、工作量计划、评审需要的基础设施和设备等，这些都需要组织管理层的支持。假如没有管理层的支持，就无法得到这些时间和资源的承诺。

（2）管理人员的培训：早期发现和修复缺陷可以节约时间和降低成本，同时管理人员必须意识到，学习新的评审技术是一项投资，其收益不是立即可见的，但随着时间的推移会越来越明显地显示出来。管理人员需要在评审成本、利益和执行方面进行有效的平衡。

（3）正规的评审过程：组织内定义并文档化评审步骤，定义不同的评审类型和评审过

程中不同的角色和职责，并对评审过程定义合适的监控手段和形式。通过评审过程的监控和改进，提供评审的度量数据（例如，评审的效率、评审发现缺陷的分布等）。

（4）对评审技术和规程进行培训：根据项目特点、评审类型，开展评审技术和规程的培训，更好地让评审参与人员了解评审的目的、评审的过程、评审的作用和意义，从而更加有效地开展评审，而不是作为过程的一部分流于形式。

（5）获得评审员和评审对象作者的支持：评审要求评审对象的作者提供合适的评审资料，并满足评审的入口准则；而对于评审员，需要他们具备合适的软件产品相关的业务技能、技术技能和评审相关知识，从而提供足够的能力完成评审工作。

（6）对最重要的文档进行评审：由于软件开发的时间和资源有限，因此，需要将评审用于最重要的文档（例如，需求、合同、计划等）。

正式而严格的评审包括 6 个阶段：计划阶段、预备会阶段、个人准备阶段、评审会议阶段、返工阶段和跟踪结果阶段。下面以“IGMP 系统需求规格说明”的评审活动（审查）为例，介绍评审过程的这 6 个通用阶段。

1．计划阶段

作者将评审对象和相关的输入材料汇聚给评审主持人，主要内容如下。

（1）评审对象：IGMP 系统需求规格说明。

（2）输入材料：IGMP 用户需求规格说明、IGMP V1 版本 RFC 1112、IGMP V2 版本 RFC 2236、IGMP V3 版本 RFC 3376。

评审主持人在评审计划阶段需要考虑以下三方面的内容。

（1）选择合适的评审员组建评审团队。选择的人员包括产品经理、产品架构师、功能架构师、测试经理、测试分析员、技术测试分析员。其中，评审的角色主要包括评审主持人（同时作为评审的记录员）、评审员和作者。

（2）评审时间和地点的安排。确定“IGMP 系统需求规格说明”的具体评审时间和评审会议召开的地点，例如，预订评审会议的会议室。

（3）评审准备时间的确定，评审对象和输入资料的分发。要求评审员在规定时间内给评审主持人反馈他们针对文档提出的建议和发现的缺陷。

假如有需要，可以在评审计划阶段确定评审的范围和重点，例如，由于评审对象的内容太多，可以选择重点部分或者风险较高部分进行评审。本次是针对“IGMP 系统需求规格说明”的第一次正式评审，因此，需要对需求规格说明的各个部分都进行仔细的检查。

评审计划阶段的信息可以通过邮件的方式分发到所有的评审员，也可以在预备会上进行详细阐述。下面是针对“IGMP 系统需求规格说明”，通过正式的邮件系统发出的“评审邀请信”。

☆示例：“IGMP 系统需求规格说明”评审邀请信

评审对象：IGMP 系统需求规格说明
输入资料：输入资料.tar
作者：XXX
评审时间：XXXXXX
评审地点：XXXXXX
下面是本次评审必须参与的评审员：
XXX（产品经理）

XXX（产品架构师）
XXX（功能架构师）
XXX（测试经理）
XXX（测试分析员）
XXX（测试技术分析员）

上述的评审员必须提交发现的缺陷或者问题，评审过程只有在所有的评审员提交comments（文档）之后才能正常结束。评审时采用的评审检查表已经包括在本邮件的附件中。

评审员提交 comments 的同时，需要提供各自花费的评审准备工作量，以方便管理工具对工作量相关的度量数据进行收集。

评审员提交的缺陷或者问题请按照缺陷过程定义的类型进行分类，以方便缺陷工具对缺陷相关度量数据的收集。

假如评审员没有任何 comments，请提交“no comments”作为一条 comments。

2．预备会阶段

发出“IGMP 系统需求规格说明”评审邀请信之后，评审主持人假如觉得有些评审的要点和注意事情还需要进行面对面的沟通，那么就有必要召开预备会。在预备会上，评审主持人会对评审的对象和相关资料进行简单的介绍，并对采用的评审技术进行简单的描述。评审主持人将“评审邀请信”中无法说清楚的一些问题和疑问进行解答，例如，建议的审查效率、至少需要的准备时间、可能采用的审查检查表等。

针对“IGMP 系统需求规格说明”实例，评审主持人认为在“评审邀请信”中已经提供足够的信息，所以省略了预备会阶段。

3．个人准备阶段

在个人准备阶段，评审员需要仔细阅读和检查评审对象，以及评审相关的输入资料。评审员根据各自的职责要求和评审的目的，尽量发现评审对象中的缺陷，并根据缺陷分类要求进行合适的分类，同时记录自己花费在检查评审对象中的准备时间和工作量。

在个人准备阶段，评审员采用合适的检查表来检查软件工作产品，是提高评审效率和覆盖率的有效手段。表 3-3 是从测试人员的角度（不包括 IGMP 相关的技术、实现等方面的内容，只包括一些通用的检查点），针对系统需求文档采用的检查表的部分内容。

表 3-3 评审检查表

序号	检 查 点	是/否	注 释
1	系统需求文档是否按照组织定义的模板编写		
2	系统需求文档是否覆盖了所有的用户需求		
3	系统需求文档是否提供了用户使用的实际场景		
4	系统需求文档是否提供了详细的参考资料		
5	需求条目是否满足可测试性要求		
6	需求条目是否满足正确性要求？例如：符合标准要求		
7	需求条目之间是否存在不一致		
8	需求条目是否标识了合理的优先级		
9	需求条目是否满足可跟踪性要求		
10	需求条目是否覆盖了非功能性特性		

个人准备阶段最主要的工作是检查评审对象，发现缺陷，并将缺陷进行合理的分类，以方便后续的缺陷分析和过程改进。表 3-4 是针对“IGMP 系统需求规格说明”采用的缺陷分类以及含义。

表 3-4　缺陷分类

缺陷分类	含　　义
缺失	评审对象内容或者条目的不全，例如：无法覆盖用户需求
多余	评审对象阐述了用户不要求的内容或者条目
模糊	评审对象内容或者条目模糊不清，没有阐述清楚
内容不一致	评审对象内容前后不一致，例如：定义的数值范围不一致
不遵循标准	评审对象内容和标准定义的不一致
错误	评审对象内容或者条目，作者对内容或者技术理解错误
编辑问题	例如：语法问题、错误的拼写等

评审员在评审会议之前，将发现的缺陷和问题，以及花费在评审准备工作上的时间和工作量分别反馈给作者和评审主持人。

4．评审会议阶段

在评审会议阶段，评审主持人应该在会议之前强调：评审的关注点是发现缺陷，而不是讨论针对缺陷的解决方案；评审的对象是软件工作产品，而不是文档的作者。假如在评审会议阶段有一些额外的议题需要讨论，也应该放在评审会议的后期进行。

评审会议阶段的一个重点工作是检查评审对象，发现和讨论其中的缺陷和问题。记录员应该详细记录缺陷的内容、分类、位置等，假如评审员之间对缺陷有不同的意见，也可以放在会议后期讨论。另一个重点工作是对个人准备阶段发现的缺陷列表进行检查，以保证缺陷列表的完整性和正确性。

在评审会议阶段，需要对评审对象给出最后的结果。例如：

（1）接受：文档不需要修改或者只要进行微小的修改。

（2）有条件接受：文档需要修改，但是不需要进一步的评审。

（3）不接受：文档需要进行深入的修改，并且需要进行重新评审或者其他的检查措施。

表 3-5 是针对“IGMP 系统需求规格说明”得到的缺陷列表的一部分。

表 3-5　IGMP 审查得到的缺陷列表

序号	缺 陷 描 述	缺 陷 分 类
1	文档中 IGMP 和 igmp 交替出现，建议统一	编辑问题
2	文档中缺少 IGMP 正式的用户使用场景	缺失
3	文档中没有详细罗列参考资料，例如：RFC 1112	缺失
4	性能指标缺乏可测试性	模糊
5	需求中定义的 IGMP 时间参数和标准不一致	不遵循标准
6	文档中缺少非功能需求，例如：可靠性、稳定性等	缺失
7	需求条目没有标识合理的优先级	缺失
8	有些需求条目在文档中前后不一致	内容不一致

根据在评审会议中汇总的 IGMP 缺陷列表，以及组织中定义的审查出口准则，评审员给“IGMP 系统需求规格说明”的评审结论为“有条件接受”。

5．返工阶段

返工阶段是作者根据评审员反馈的缺陷列表和建议对文档内容进行修改。假如评审的最后结果是不接受，那么作者除了进行修改之外，还需要为下一个阶段的再次评审做好相关的准备工作。

“IGMP 系统需求规格说明”的评审结论是“有条件接受”，因此，文档作者必须对评审过程中发现的缺陷进行修复和更新，例如，增加 IGMP 功能的非功能性需求条目、规范 IGMP 术语等。

6．跟踪结果阶段

跟踪结果阶段主要是评审主持人或者其他指定人员对返工阶段的输出进行检查，验证作者是否正确完成了修改任务。针对“IGMP 系统需求规格说明”，评审主持人指定其中的一位评审员：测试分析员来对作者修改之后的文档进行跟踪和检查，并对修改之后的文档给出结果建议。

跟踪阶段的另一个重要工作是对“IGMP 系统需求规格说明”评审过程中的一些数据和信息进行收集和分析，例如，分析 IGMP 系统需求规格说明的质量、生成 IGMP 系统需求规格说明过程的有效性、审查过程本身的有效性和效率。评审过程中收集的数据包括：被评审的软件工作产品、评审召开的日期、参与评审的成员、评审员准备的时间、评审会议的时间、评审对象规模、评审结果等，它们可以用来分析评审质量和效率。同时，跟踪阶段也需要不断优化和改进检查表的质量。

☆示例：IGMP 系统需求规格说明审查收集的数据

如下所示是针对“IGMP 系统需求规格说明”审查收集的主要数据。

“IGMP 系统需求规格说明”审查收集的数据

1. 评审基本资料：

   评审对象：IGMP 系统需求规格说明
   输入资料：输入资料.tar
   作者：XXX
   评审时间：XXXXXX
   评审地点：XXXXXX

2. 实际评审参与人员：

   XXX（产品经理）
   XXX（产品架构师）
   XXX（功能架构师）
   XXX（测试经理）
   XXX（测试分析员）
   XXX（测试技术分析员）
   XXX（客户代表）
   XXX（过程改进组成员）

3. 评审员准备时间：

   XXX（产品经理）：4 小时

XXX（产品架构师）：3 小时
XXX（功能架构师）：3.5 小时
XXX（测试经理）：3 小时
XXX（测试分析员）：4 小时
XXX（测试技术分析员）：4 小时
XXX（客户代表）：3 小时
XXX（过程改进组成员）：2 小时
4. 评审会议持续时间：2 小时 15 分钟
5. 评审对象规模：8 页
6. 评审发现的缺陷列表：参见表 3-5。
7. 评审结果：有条件接受

3.3 对评审进行管理

评审是静态测试的重要组成部分，它是通过直接检查软件工作产品来发现缺陷的。评审通常发现的是软件工作产品的缺陷，而非失效。软件工作产品可以是软件开发生命周期中的各种文档，例如，需求规格说明、设计规格说明、测试计划、测试设计规格说明等。了解评审的基本原则、评审相关角色和职责、评审类型、评审影响因素等有助于更好地开展评审活动。

3.3.1 评审基本原则

为了有效地开展评审活动，需要理解和掌握一些评审的基本原则，以提高评审效率和有效性，并避免评审过程中的一些误区。

1. 尽早开展评审活动

评审对象（例如，软件工作产品或者过程）满足评审入口准则（例如，必须遵循的标准）之后，就可以开展评审活动。测试人员积极参与评审活动，就是尽早参与软件开发生命周期的主要手段，不仅可以尽早发现软件工作产品中的缺陷，大大减少缺陷修复的时间和成本；同时可以尽早熟悉被测对象的业务和技术要求，从而更好地指导和规划后续的测试活动。

2. 控制评审会议的时间

通常，评审会议应该控制在两个小时之内。如果需要，可以在当天再发起另外一个评审会议。时间过长的评审会议会导致评审人员容易疲劳，从而降低评审的效率和有效性。

3. 评审关注的对象不是作者

评审关注的是软件工作产品或者过程，而不是作者，因此评审员必须要注意他们在评审过程中的言语以及表达方式，例如，缺陷的描述方式、评审会议讨论问题的语气等，避免作者成为被攻击目标或者被迫处于防御的位置。评审员提交的问题不应该以命令或者嘲笑的形式写给作者，中肯的改进或者修改建议对软件工作产品的质量改进是有帮助的，并

且是明智的。同时，每个评审员都必须有机会充分表达他们各自的观点，并且评审会议纪要必须完整地记录每个评审员的意见和建议。

4. 发现缺陷而不是修复缺陷

评审会议阶段应该将关注点放在尽量多的发现被评审对象中的缺陷上面，而尽量避免讨论针对缺陷的可能的解决方案和方法。提出解决方案以及对应的讨论不应该是评审会议的关注点。评审过程中发现的缺陷和问题应该划分为不同的严重程度级别。例如：

（1）严重缺陷：评审对象不能满足它的目标，在批准评审对象之前必须修正相关缺陷。

（2）重要缺陷：影响评审对象的可用性，批准评审对象之前应该修改相关缺陷。

（3）一般缺陷：小的偏差，基本不影响使用。

（4）没有缺陷：返工时无须修改。

5. 最后的评审意见

评审团队应该对评审对象给出最后的评审意见。例如，接受、有条件接受和不接受（请参考 3.2.3 节的内容）。

3.3.2 评审影响因素

软件开发生命周期的每个阶段结束点或者里程碑点，通常都需要对该阶段的输出开展评审活动，例如，系统需求规格和设计规格定义之后再进行评审。评审以软件项目的商业目标为起始点，直到软件设计不再可分的层次，因此评审可以看成是自顶向下进行的。而管理评审是在软件项目的主要里程碑点进行，通常作为测试执行或者其他主要项目阶段之前、过程中和之后的验证活动的一部分。

评审策略必须与测试方针和总体测试策略保持一致。测试经理在制定软件项目的总体评审计划之前，需要得到评审主持人（评审主持人也可能就是测试经理）支持，至少需要考虑下面的几个因素。

（1）评审对象：主要关注在软件工作产品还是过程。根据不同的对象，需要采用不同的评审类型，例如，假如关注过程，那更可能采用管理评审。

（2）评审参与人：不同的评审对象，需要参与的评审人员是不一样的，例如，关注软件工作产品的评审人员，通常是关注技术层面较多；而管理评审，更多的参与人是经理，更关注在项目状态和进度等方面。

（3）需要覆盖的相关风险因素：包括软件工作产品技术相关的风险，同时包括组织人员、技能、时间等方面的风险。

评审主持人在项目计划初期，必须确保评审过程的目标、如何进行有效和高效地开展评审，以及评审反馈等方面达成共识。同时需要识别评审对象，并选择适当的评审类型（针对软件工作产品，可以选择非正式评审、走查、技术评审或者审查，或者是各种评审类型的混合），以及评审过程的正式程度。另外，评审主持人应在计划阶段充分定义用以评估评审的度量。假如采用审查评审类型，评审人员应该在软件工作产品的单个需求或者章节方面，按照作者的要求进行简要审查。不管是针对整个系统进行评审，还是针对子系统甚至单个软件进行评审，都需要根据项目规模和复杂度以及产品风险决定评审的次数、评审的类型、评审组织以及涉及的评审人员。

为了达到更好的评审效果，在软件项目团队中提供评审过程、评审角色和职责、评审类型等方面的专门培训，以确保评审人员了解他们在评审过程中的角色和职责。为了保证效率，评审参与人必须有足够的评审对象相关的业务和技术技能，以及评审过程相关的技术和规程方面的知识，除了这些，为了让评审有效，其他的一些技能对于评审员也是重要的，例如，严谨和关注细节。好的评审意见必须足够清晰且按照恰当的优先级排列。

测试经理在确定上述内容之后，测试经理可以为评审过程分配时间和资源预算。确定评审的预算过程，测试经理必须考虑风险评估和投资回报率的计算。评审的投资回报率指的是开展评审的成本，与不进行评审而导致在项目后期才应对相同的缺陷的成本之间的区别（或完全遗漏这些缺陷到用户现场而导致的成本）。通过计算采用评审可能节省的时间和成本，具体的计算方法可以参考 2.7 节的质量成本计算方法，并确定具体的数值。具体确定执行评审的最佳时机，依赖于下面几个方面的因素。

（1）待评审对象是否有足够确定的版本可用；

（2）具备合格技能的评审参与人是否有时间参与；

（3）最终版的评审对象应该到位的时间；

（4）特定评审对象的评审过程所需时间。

制定评审计划时，测试经理需要考虑评审过程中涉及的技术、组织和人员相关的风险。具备足够业务和技术技能的评审人员的积极参与，是评审成功的关键因素。软件项目的每个团队都应该参与评审计划的制定，以确保所有团队都对评审过程的成功做出承诺。评审计划必须确保每个组织为评审人员分配足够的评审准备和参与的时间，他们能够按照项目进度安排的评审时间节点参与评审。测试经理还必须识别关键评审人员的备选成员，以防关键评审人员因为个人原因或工作原因无法参与评审。

评审计划时有效应对技术、组织和人员相关的评审风险，可以提高评审的效率和有效性，下面是其中的一些建议。

1．技术因素

（1）保证正确遵循针对评审类型所定义的评审过程，特别是针对正式的评审，例如，审查；

（2）记录评审所花费的成本（例如：时间成本）和所获得的收益；

（3）尽早参与文档的评审，例如，文档的初稿阶段，从而在早期识别文档中的缺陷，防止它们被引入到后续的文档中；

（4）启动评审之前，检查评审计划中定义的评审入口准则，以确保评审对象已为开展评审准备就绪；

（5）运用组织特有的缺陷检查表提高评审的效率和有效性；

（6）根据不同的目标（例如，技术改进、信息转移或进度管理），运用多种类型的评审；

（7）针对影响重大决策的文档进行评审时，例如，在决定是否批准项目主要预算之前，需要认真检查其中的合同条款、建议或要求；

（8）在时间和资源有限时，可以对评审对象进行抽样调查，以达到评估的目的；

（9）评审应该注重内容而非形式，鼓励发现最重要的缺陷；

（10）持续改进评审过程。

2. 组织因素

（1）即使软件产品发布有最后期限的压力，管理人员也应该确保花费足够的工作量用于评审活动；

（2）切记评审中花费的工作量和成本并非和发现的缺陷数目成比例；

（3）对于在评审中发现的缺陷，要给予足够的时间进行修改；

（4）永远不要将评审中的度量数据用于个人绩效评估；

（5）对于不同类型的评审，要确保能有合适的评审员参与；

（6）为评审参与人员提供评审方面的培训，特别是针对正式的评审类型；

（7）成立评审主持人论坛相互分享经验和想法；

（8）确保人人参与评审，并且保证每个人都对自己负责的文档内容进行了评审；

（9）将最正式的评审技术用于最重要的文档；

（10）对于由不同技术和背景的人员组成的评审团队，要确保其具有良好的平衡性；

（11）对通过评审过程所取得的改进表示认可。

3. 人员问题

（1）使项目利益干系人认识到，评审将会发现缺陷，并改进软件工作产品的质量；

（2）对于缺陷修复和再评审要给予充足的时间；

（3）要确保评审对于作者而言是正面的、积极的经历；

（4）营造一种“无责备”的氛围，从而乐于接受缺陷的识别；

（5）要确保评审意见具有建设性、有益性和客观性，而非主观性；

（6）作者不同意或者不愿意的情况下，不进行评审；

（7）鼓励大家对评审文档中最重要的方面进行深层次思考。

评审主持人是成功实施评审的关键角色。除了支持测试经理制定评审计划外，评审主持人在实际执行正式评审时，评审主持人必须确保：

（1）评审参与者提供的评审相关度量数据，有助于评估评审效率和有效性；

（2）针对不同的评审对象编制了合适的评审检查表，并根据评审反馈信息对检查表进行维护，以改进将来的评审效果；

（3）评审过程中发现的缺陷，需要明确定义缺陷严重程度和优先级，以方便对评审进行评估，同时可以用于缺陷管理（内容请参考第 4 章）。

完成评审会议阶段之后，评审主持人还需要对结果进行跟踪，并对后续的过程改进提供信息。简单而言，评审主持人在完成每次评审会议之后，还需要关注：

（1）确保评审过程中识别的缺陷得到了充分的解决，评审结果符合评审计划中确定的特定测试目标；

（2）收集评审过程中的各种度量信息，并以此为基础确定最终的评审投资回报率 ROI；

（3）根据收集的度量信息向软件项目利益干系人提供反馈信息；

（4）同时，向评审参与者提供反馈信息。

为了评估评审的有效性，测试经理可以对评审之后的动态测试实际结果与评审报告上的结果进行比较。假如经过评审之后的软件工作产品，其在动态测试过程中依旧发现较多的缺陷，评审主持人应该考虑评审过程中什么地方会遗漏缺陷，为什么会遗漏这些缺陷的问题。可能导致评审遗漏发现缺陷的原因是多方面的，例如，评审出口准则没有严格执行、

评审参与人员选择不当和经验不足、没有合适的评审检查列表、评审准备时间不足和评审会议太短、评审人员对评审重视程度不够等。

假如评审阶段经常出现将严重缺陷遗漏到后续测试过程中，说明评审工作的开展存在严重的问题，这时评审主持人需要检查评审过程并采取适当的措施。评审有效性的降低，必定是各种原因造成的。在软件项目回顾会议中需要识别这些根本原因并在后续项目评审中得到持续的改进。

不管是评估评审有效性，还是评估评审质量，都需要合适的度量数据的支持。3.4 节将具体列出针对软件工作产品和评审过程的度量。测试经理需要注意的是：不管什么情况，评审度量数据都不应该用于惩罚或者奖励评审人员或者文档作者，而是应该关注于评审过程。

3.4 评审度量

评审过程中收集度量数据，其主要目的是为了评估被评审工作产品的质量和评审过程的有效性，以判断通过评审之后工作产品质量是否得到了显著的改善，以及改进过程活动。评审过程中的这些度量信息也将帮助确定评审所带来的收益，例如，确定评审的投资回报率。为了达到评审目的，评审主持人必须确保有可用的度量用于：

（1）评估评审对象的质量水平；

（2）评估进行评审的成本；

（3）评估进行评审后的下游收益。

针对软件工作产品评审的监控和评估，可以测量和报告的度量类型有很多，具体选择哪些度量，依赖于不同的度量目标。下面是针对软件工作产品的部分度量列表。

（1）软件工作产品规模，例如，代码行、文档页数等；

（2）评审人员评审前的准备时间；

（3）评审会议的持续时间；

（4）修复缺陷的返工时间；

（5）评审过程的持续时间；

（6）评审过程中发现缺陷数目和缺陷严重程度分布；

（7）识别软件工作产品中的缺陷集群，即缺陷在不同工作产品或者功能模块中的分布；

（8）采用的评审类型，例如，非正式评审、走查、技术评审、审查还是管理评审；

（9）平均缺陷密度，例如，每千行代码的缺陷数目，或每页文档的缺陷数目；

（10）估计的遗留缺陷数目，或遗留缺陷密度。

针对评审过程的评估，同样存在有不同的度量类型。具体采用什么度量依赖于过程的度量目标。下面是部分针对评审过程的度量列表。

（1）评审阶段的缺陷发现百分比；

（2）评审过程工作量和成本的改进程度；

（3）计划评审的软件工作产品的覆盖率百分比；

（4）发现的缺陷类型和其严重程度分布；

（5）评审参与人对评审过程有效性和效率的反馈；

（6）不同阶段发现缺陷的质量成本度量比较：评审阶段、动态测试阶段和软件产品运行阶段；

（7）评审有效性的评估，例如，评审类型与缺陷发现有效性之间的相关性评估；

（8）参与评审的人员数目；

（9）单位工作小时发现的缺陷数目；

（10）通过评审估计可以节省的项目时间；

（11）平均缺陷工作量，即发现和修复缺陷的总时间除以缺陷数量得到的结果。

测试经理必须深入理解上面提到的各种度量，不管是用于评估软件工作产品的度量，还是用于评估过程的度量。但记住，尽量避免将评审过程中的度量用于个人能力或者绩效的评估。对于测试经理而言，基于测试过程中的度量数据对测试人员进行奖励和惩罚，都是一个糟糕的管理实践，特别是测试人员无法控制这些度量数据的情况下。另外，上述提到的用于评估产品质量的度量，同样也可以用于过程评估。

3.5 管理正式评审

ISTQB 基础级大纲描述了不同的评审类型，包括审查、技术评审、走查和非正式评审，以及本大纲中描述的管理评审和审计，具体的内容参见 3.2 节。不管采用什么评审类型，其正式的评审过程都由不同的阶段组成，并有不同的角色参与。根据 IEEE Std 1028—2008 的定义，评审的正式过程由以下 6 个阶段组成。

（1）计划阶段：选择评审员，分配角色；为更加正式的评审类型（例如审查）规定评审的入口准则和出口准则；选择需要进行评审的文档或文档章节等。

（2）预备会阶段：分发文档，向评审参与者解释评审的目标、过程和文档；核对入口准则（针对更正式的评审类型）。

（3）个人准备阶段：在评审会议之前，每位评审参与者准备各自的评审工作，标注评审对象中可能的缺陷、问题和建议。

（4）评审会议阶段：讨论评审员提交的问题列表，并形成会议纪要（针对更正式的评审类型）。会议参与者可以标识缺陷、提出处理缺陷的建议或对缺陷做出决定。

（5）返工阶段：修复评审过程中发现的缺陷，通常由作者进行。

（6）跟踪结果阶段：检查缺陷是否已解决，收集度量数据，并评估出口准则（针对更正式的评审类型）。

评审主持人想要正确实施正式评审，必须确保评审过程遵循了上述的所有 6 大阶段，同时必须定义合适的角色参与并在其中承担相应的职责。正式评审有许多特点，例如：

（1）定义评审的入口准则和出口准则。评审入口准则指的是满足哪些条件可以开始评审工作；而出口准则指的是何时可以结束评审活动。入口准则和出口准则都可以和评审过程中的工作产品度量和过程度量结合起来。

（2）定义评审检查表。针对软件工作产品的检查表可以有效地提高评审有效性，例如，针对测试用例规格说明的检查表，有助于提高评审有效性。

（3）定义评审交付物，包括评审会议纪要、针对软件工作产品质量的评估表，或其他评审总结表等。

（4）定义评审相关的软件工作产品度量和过程度量。正式评审会采用各种度量来报告评审的效率、有效性和进度等结果。

评审主持人在发起正式评审之前，应该首先检查是否满足了评审的入口准则。评审入口准则中定义的各项要求应该在软件项目相关利益干系人之间达成一致。假如评审入口准则没有满足，评审主持人可以向评审决策者提出建议，例如：

（1）修正评审目标，并对评审过程重新定义；

（2）为了使评审继续，进行必要的修正活动；

（3）推迟评审。

最后，评审活动是软件开发生命周期的重要组成部分，因此正式评审应该在整体项目环境下得到监控，并与软件项目的质量保证活动协调一致。根据软件工作产品度量和过程度量，对开发过程、测试过程、评审过程和管理过程等提供反馈信息，都属于正式评审的控制手段。

小结

评审是静态测试的重要组成部分，可以在软件开发生命周期的任何阶段实施。评审是测试尽早介入基本原则的一个体现，其对象既可以是开发过程中的各种交付物（软件工作产品），也可以是针对过程。除了 ISTQB 基础级大纲中涉及的主要针对软件工作产品的非正式评审、走查、技术评审和审查之外，也包含主要针对过程的管理评审和审计。

管理评审和审计主要的关注点是过程，因此其主要目的不是发现缺陷。管理评审的主要目的是监控项目过程、确定项目计划和执行进度状态，或者评估为获取某个目的而采用的管理方法的有效性。管理评审可以帮助确定合适的纠正措施、项目资源分配的变更，或者项目范围的变更。而审计的主要目的是提供独立的评估，证明工作产品和过程与规范、标准、指南、计划、规格说明和规程等的一致性。

评审的成功与否会受到各种因素的影响，因此评审主持人应该确保当前的环境条件与 ISTQB 基础级大纲中定义的确保评审成功实施的因素相符合。同时，具备特定产品业务技能和技术技能的利益干系人参与评审是评审成功实施的重要条件。而测试人员通常同时具备针对测试对象的业务技能和技术技能，例如，了解用户使用软件产品的操作行为和理解该软件产品所必须具备的功能特性等，因此他们积极参与评审过程也是非常重要的。

评审主持人需要在评审过程中收集不同的度量信息，包括针对软件工作产品的度量和针对过程的度量，其主要目的是评估它们的有效性，以判断后续的工作产品质量和过程质量是否得到了明显的改善。同时，收集和分析这些度量信息也有助于确定评审能带来的收益，以评估评审的投资回报率。

正式的评审过程必定会遵循评审的 6 大阶段：计划阶段、预备会阶段、个人准备阶段、评审会议阶段、返工阶段和跟踪结果阶段，同时每个阶段有不同的角色参与并承担不同的职责。评审过程不仅应该和软件项目的质量保证活动协调一致，同时需要对开发过程、测试过程、管理过程等提供有效反馈信息，从而持续改进过程能力。

模拟题

31．CTAL-ATM_LO-3.2.1

TM-3.2.1(K2)了解管理评审和审计的关键特征。

问题：

作为测试经理，你与其他项目管理团队的成员一起参加会议。会议议程是根据系统测试出口准则、验收测试入口准则以及其他业务考虑，讨论是否可以开始验收测试。下面哪个论述正确？

选项：

A．该会议属于管理评审，因为项目管理团队在评估当前的状况并确定下一步行动

B．该会议属于审计，因为项目管理团队通过检查证据，检查与已定义准则之间的一致性

C．该会议属于管理评审，因为项目管理团队执行检查以确保能够达到项目进度要求

D．该会议属于审计，因为项目管理团队将要对准则进行通过/失败的评估

解释：

A．正确。利用测试出口和入口准则评估状态，并基于评估结果以确定后续的行动，该行为在大纲中属于管理评审的范畴。

B．不正确。确实在检查与已定义准则之间的差异，但是没有对一致性进行独立评估（该检查由项目团队完成），同时并没有表明在检查这方面的“证据”。

C．不正确。该论述在一定程度上是正确的，但它没有考虑管理层做出的决策，这是决定属于管理评审还是审计的关键理由。

D．不正确。同选项 B 一样，虽然可能存在对准则进行通过/失败的评估，但是没有对一致性进行独立评估。

分值：1 分

32．CTAL-ATM_LO-3.3.1

TM-3.3.1(K4) 分析项目，选择合适的评审类型并定义开展评审的计划，确保评审得到适当的实施、跟进、确定责任。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

作为项目的一部分，假设高级管理团队要求你计划评审活动。他们希望采用轻量级的过程，及早发现一些缺陷，同时在团队之间达成共识和一致的理解。

下面哪一个是描述了当前情况下最好的选项？

选项：

A. 你应该计划非正式评审，并针对所有合适的工作产品选择合适的评审参与者

B. 你应该计划需求、设计和代码的审查

C. 你应该计划为产品风险分析、测试用例和测试计划进行非正式评审

D. 你应该说服管理层让测试经理以外的人来计划评审活动

解释：

A. 正确。非正式评审属于轻量级方法，可以达到预期收益。

B. 不正确。管理层希望采用轻量级过程，而且需求已经完成（可能设计也已完成）。

C. 不正确。选项 C 没有选项 A 好，因为选项 C 只包含测试工作产品。

D. 不正确。评审可以由项目中的各类参与者进行计划和管理，包括测试经理。

分值：3 分

33. CTAL-ATM_LO-3.3.2

TM-3.3.2(K2)了解参与评审需要的因素、技能和时间。

问题：

场景：

假设你为某个刚启动的项目工作。该项目构建可以定制积分忠诚度与奖励计划的系统，专门为小企业基于 Web 上进行销售的中小规模公司提供个性化的积分与奖励系统，中等规模的企业可以在他们的 Web 上进行销售。这些公司将他们自己的信息注册到该系统的 Web 数据库。系统允许这些公司创建定制化的按钮，放在他们自己的网站，从而允许客户将他们的信息登记到公司的积分忠诚度与奖励计划中。后续的每次购买活动都可以获得积分，公司和他们的客户都可以对该计划进行管理，例如，企业可以确定免费为客户提供商品和服务所需的积分点数，而客户可以了解自己的积分情况，确定获得免费产品或者服务所需的积分。

公司的销售人员对该系统进行了大幅度的促销，针对第一批最早注册的公司提供了第一年费用的大幅折扣。销售材料中说明，针对公司和客户，该服务都可以提供高可靠性和极快的响应速度。

在目前阶段，需求规格说明已经完成，软件开发工作刚刚开始。当前的进度计划将允许中小公司和他们的客户在三个月之后开始注册登记。

公司准备使用云计算资源开展该服务，除了普通的办公计算机之外，不再为开发人员、测试人员、其他工程师和经理提供额外的硬件资源。该系统的构建将使用行业标准的基于 Web 的应用软件组件。

假设高级管理层要求你在该项目中对评审进行管理。你正在为评审识别的质量风险项选择评审人员。

为了有效开展评审活动，考虑下面所需的能力；

Ⅰ 技术技能

Ⅱ 合适的个性特点

Ⅲ 过程知识

Ⅳ 业务知识

考虑下面某个员工的概述，他可能会参加评审活动：

1 有以前金融应用的测试经验

2 具备简单的 Web 应用开发经验

3 有丰富的评审经验
4 关注细节
5 了解云计算
下面哪个选项正确地匹配了员工细节与所需技能？

选项：

A．I-2; I-5; II-4; III-3; IV-1
B．I-1; I-2; II-4; III-3; IV-5
C．I-4; II-2; II-3; III-5; IV-1
D．I-2; II-3; III-4; III-1; IV-5

解释：

A．正确。Web 开发经验和了解云计算与项目的技术技能相关。关注细节是任何评审参与者所需具备的个性特点。参加过评审活动可以为评审参与者提供评审过程的知识。金融应用可以用来管理账目平衡，这与管理积分和奖励平衡相关。
B．不正确。有一个或者多个不匹配。
C．不正确。有一个或者多个不匹配。
D．不正确。有一个或者多个不匹配。
分值：1 分

34．CTAL-ATM_LO-3.4.1

TM-3.4.1(K3)定义评审使用的过程和产品度量。

问题：

你在国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

你的产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线的目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年的工作时间，大部分人具有更长的工作时间。经验丰富的测试人员利用公司内部的定制测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态，对所有业务单元个人效率进行度量。公司在公司级别实施了 CMMi。

团队在紧跟硬件开发进度方面存在问题。你的经理认为，如果测试人员参与业务需求的评审，则可以在项目中更高效地发现一些缺陷。下面哪三个度量可以最好地证明使用评审可以达到此目的？选择最佳的三个选项。

选项：

A．动态测试中发现的缺陷数目
B．动态测试覆盖率
C．评审和动态测试小时数
D．评审中发现的缺陷数目
E．动态测试中发现的严重缺陷数
F．测试执行状态
G．硬件业务团队和软件业务团队的评审结果

解释：

A．正确。因为根据这些度量可以计算评审和动态测试花费的总时间，以及它们各自发现的缺陷数，再将这些数据与动态测试的数据进行比较。
B．不正确。它与效率计算无关。
C．正确。
D．正确。
E．不正确。该场景中明确提到，你可以得到该数据。仅计算严重缺陷不合适。
F．不正确。该场景中明确提到，你可以得到该数据。但是考虑状态不合适。需要测试的小时数。
G．不正确。该选项听起来比 D 选项好一些。但即使评审结果是缺陷，也不能包含硬件评审，因为问题的意思是你希望通过评审发现软件需求的缺陷，而这些可能会在将来变成缺陷而被动态测试发现。

分值：2 分

35．CTAL-ATM_LO-3.5.1

TM-3.5.1(K2)举例说明正式评审的特点。

问题：

你是信息应用（App）敏捷开发项目的测试经理。由于用户提供的功能不全或者不正确，项目团队计划评审所有的用户故事。评审将由你（测试经理）来领导。评审的主要目的是所有项目干系人在用户故事格式、颗粒度、完整性和准确性方面达成一致。下面的人员角色将作为评审员参与评审：核心开发人员（CD）、测试分析师（TA）、产品

经理（PM）和领域专家（DE）。在预备会阶段，核心开发人员（CD）抱怨评审活动影响了他自己的工作。

个人准备阶段之后，表 3-6 描述了 4 位评审员各自发现的缺陷数目。

表 3-6　发现的缺陷数

	CD	TA	PM	DE
主要	2	8	6	5
次要	2	11	5	7
拼写错误	8	14	9	11

你需要决定如何继续开展评审活动。

下面哪个选项是测试经理应该选择的?

选项:

A．要求 TM 和 CD 讨论一下，是否通过邀请其他成员介入减轻 CD 的工作量

B．推迟评审进度，重新定义 CD 的评审目标

C．加快评审进度，将 CD 指定为记录员的角色

D．取消该评审，提交报告给上层经理，说明 CD 无法介入评审的问题

解释：

A．正确。可以通过邀请其他成员介入以减轻 CD 的工作量，同时保证评审的顺利进行。

B．不正确。所有项目干系人都必须统一评审目标。

C．不正确。惩罚没有任何意义，应该采用建设性的合作。

D．不正确。抱怨不解决问题，应该采用建设性的合作。

分值：1 分

第4章 缺陷管理

本章学习目标如表 4-1 所示。

表 4-1 学习目标

编　号	学习目标描述	级　别
TM-4.2.1	为测试组织开发缺陷管理过程，包括缺陷报告流程，用于在测试生命周期中监控项目缺陷	K3
TM-4.2.2	说明有效的缺陷管理必需的过程和参与者	K2
TM-4.3.1	定义在缺陷管理过程中应该收集的数据和分类信息	K3
TM-4.4.1	解释缺陷报告信息怎样用于评估测试和软件开发过程的过程能力	K2

相关术语如表 4-2 所示。

表 4-2 术语

英　文	中　文	说　明
anomaly	异常	与基于需求文档、设计文档、用户文档、标准或用户期望和经验所得出的预期之间出现的任何偏差情况，都可称为异常。异常可在且不限于在下面的过程中被识别：评审、测试分析、编译、软件产品或应用文档的使用等情形。参见 bug，defect，deviation，error，fault，failure，incident，problem
defect	缺陷	组件或系统中会导致组件或系统无法执行其必需功能的瑕疵，例如，错误的语句或变量定义。如果在组件或系统运行中遇到缺陷，可能会导致失效[GBT 11457]
defect triage committee	缺陷分类委员会	利益干系人的跨职能团队，管理从初始发现到最终解决（缺陷移除、缺陷延期或报告取消）的已报告的缺陷。某些情况下，与配置管理委员会是相同的团队。参见 configuration control board
failure	失效	组件/系统与预期的交付、服务或结果存在的偏差（与 Fenton 一致）[GBT 11457]
false-negative	假阴性结果	测试结果未能识别出测试对象中的缺陷
false-positive	假阳性结果	测试结果中报告了测试对象实际不存在的缺陷
phase containment	阶段遏制	缺陷，在软件生命周期中的某个阶段引入，并在同一个阶段被移除所占的百分比
priority	优先级	赋予某项（业务）重要性的级别。例如，缺陷级别[GBT 11457]
root cause	根本原因	指缺陷产生的根源。如果清除了一个缺陷的根源，那么该缺陷即被清除了[CMMI]
severity	严重程度	缺陷对组件/系统的开发或运行造成的影响程度（与 IEEE 610 一致）[GBT 11457 严重性]

4.1 简介

测试是软件开发生命周期中的重要组成部分，其主要目的是尽早和尽量多地发现软件产品中存在的缺陷，协助开发人员更好地定位和修复缺陷，并针对缺陷修复所做的变更进行确认测试和回归测试。有效开展测试活动可以作为提高软件产品质量、降低成本和预防缺陷的重要手段。

缺陷作为测试过程中的一个重要输出，其在评估和改进软件产品质量、提高测试效率和有效性、持续改进开发过程和测试过程等方面具有重要的意义。软件开发生命周期中提交、分析和监控缺陷的主要目的和作用可以表现在多个方面，例如：

（1）为开发人员和其他人员提供问题反馈，帮助他们定位、隔离和修复缺陷；

（2）为项目管理人员提供被测对象的质量评估和进度监控所需的度量信息，并基于度量信息做出合理的决策；

（3）为测试过程改进和开发过程改进提供有用的数据和信息。

测试过程中每个测试人员都应该积极参与缺陷管理过程，为实现上述目标提供所需的度量信息。测试经理主要关注缺陷的评估、跟踪和监控等活动，而测试分析师和技术测试分析师主要关注如何在测试过程中尽量多地识别缺陷，并进行正确的记录，以及后续的缺陷验证和回归测试工作。

测试过程中发现的缺陷，测试经理需要从缺陷发现、缺陷提交、缺陷分类、缺陷修复和解决方案验证等整个过程进行跟踪。为了有效且及时地解决测试过程中发现的缺陷，组织内需要建立一套完整的过程和规则对缺陷进行跟踪和管理。另外，测试经理还必须熟悉哪些缺陷相关的数据是需要及时收集和分析的，此时合理的缺陷管理过程和缺陷管理工具是必需的。详细的缺陷报告内容，可以参考 IEEE Std 829—2008 标准。

4.2 缺陷生命周期和软件开发生命周期

软件测试贯穿于整个软件开发生命周期，开发过程中的所有工作产品都可作为测试对象，例如，需求规格说明、用户故事、技术文档、测试用例、程序代码等。软件开发生命周期中的任何阶段和任何工作产品都可能引入缺陷，因此任何阶段都应该针对该阶段输出的工作产品开展测试活动以发现缺陷，并推动开发人员尽快修复缺陷。

软件开发生命周期的不同阶段会引入不同的缺陷，因此每个阶段需要采用不同的测试技术和方法识别和移除本阶段存在的缺陷失效，包括静态测试和动态测试。例如，软件开发生命周期的早期，针对设计规格说明和代码进行静态评审，有助于尽早发现该阶段存在的缺陷，并更早地推动开发人员修复缺陷，从而降低质量成本；而单元测试、集成测试和系统测试等动态测试过程中，可以发现由于被测对象中存在缺陷而导致的失效，即测试得到的实际结果与期望结果不一样。为了简化描述，本章中的缺陷泛指软件开发生命周期中出现的缺陷、异常和失效等。

缺陷可以在不同阶段引入，也可以在不同阶段被发现。假如缺陷引入阶段和缺陷发现阶段相同，就可以有效实现缺陷的阶段抑制能力，即缺陷引入阶段就识别和修复缺陷，此时质量成本是最低的。测试的尽早介入有助于实现缺陷的阶段抑制能力，此时通过静态测试发现的是缺陷而不是失效，因此不需要通过调试就可以移除缺陷，从而降低缺陷修复的成本。

假如软件开发生命周期采用测试驱动开发的方式，此时可以将单元测试用例看作是一种可执行的设计规格说明，代码开发后可立即执行这些测试。假如测试执行的结果是没有通过，此单元的开发就不能算完成。这类测试发现的缺陷，通常不会以缺陷报告的形式提交，而是通过不断地对代码进行增强和重构以达到通过所有测试的目的。

4.2.1 缺陷工作流程和状态

和软件开发生命周期一样，缺陷管理过程也是由一系列的阶段和活动组成的，即缺陷管理同样具有生命周期。如图 4-1 所示，根据 IEEE Std 1044—1993[①]异常管理标准（本文中用缺陷代替标准中的异常）的描述，缺陷生命周期主要由 4 个阶段组成：识别（Recognition）、调查（Investigation）、改正（Action）、总结（Disposition）。

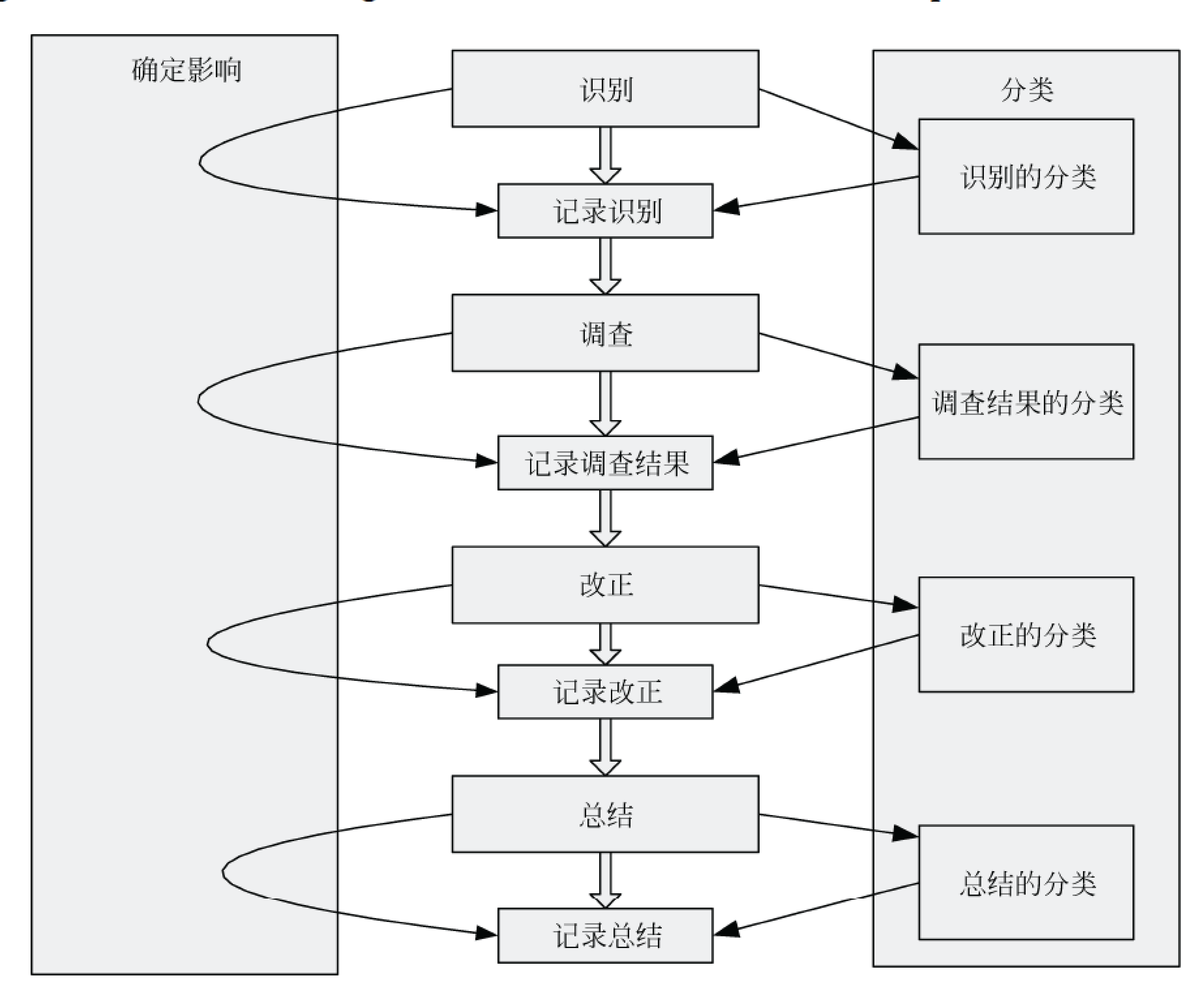


图 4-1 缺陷分类过程

缺陷生命周期的每个阶段都包括记录（Recording）、分类（Classifying）和确定影响

① IEEE 1044—1993 对应的国标是 QJ 3026—1998。

（Identifying Impact）三个活动。缺陷生命周期的 4 个阶段看起来是按照顺序进行的，但是缺陷可能会在这几个阶段中进行多次迭代。下面对缺陷生命周期的每个阶段和阶段中的活动进行详细的讨论。

1．识别

缺陷识别可以发生在软件开发生命周期的任何一个阶段。缺陷的识别可以由参与项目的任何利益干系人完成，例如，系统人员、开发人员、测试人员、支持人员、用户等。缺陷识别阶段的主要活动如下。

（1）记录：在缺陷识别阶段，需要记录缺陷的相关信息，包括发现缺陷时的支持数据信息和环境配置信息，例如，被测系统的硬件信息、软件信息、数据库信息和平台信息等。

（2）分类：在缺陷识别阶段，需要对缺陷相关的一些重要属性进行分类，主要包括发现缺陷时执行的项目活动（如表 4-3 所示）、引起缺陷的原因、缺陷是否可以重现、缺陷发现时的系统状态、缺陷发生时的征兆等。

表 4-3　发现缺陷时的项目活动分类

类　别	符合度要求	代　号	分　类
项目活动 RR100	强制性	RR110	分析
	强制性	RR120	评审
	强制性	RR130	审计
	强制性	RR140	审查
	强制性	RR150	编码/编译/汇编
	强制性	RR160	测试
	强制性	RR170	确认测试/鉴定测试
	强制性	RR180	支持/操作
	强制性	RR190	走查

（3）确定影响：根据缺陷发现人员的经验和预期，判断缺陷可能会造成的影响，例如，缺陷的严重程度（如表 4-4 所示）、优先级，以及缺陷对成本、进度、风险、可靠性、质量等的影响。

表 4-4　严重程度分类

类　别	符合度要求	代　号	分　类
严重程度 IM100	强制性	IM110	危急
	强制性	IM120	高
	强制性	IM130	中
	强制性	IM140	低
	强制性	IM150	无

2．调查

需要对每个识别的缺陷进行调查。调查阶段主要是用来发现可能存在的其他问题以及相关的解决方案，解决方案包括“不采取任何行动”。缺陷调查阶段的主要活动如下。

（1）记录：在缺陷调查阶段，需要记录相关的数据和信息，对缺陷识别阶段记录的信息进行更新。缺陷调查阶段记录的信息包括缺陷调查者的信息、缺陷调查的计划开始时间、计划结束时间、实际开始时间、实际结束时间、调查工作量等。

（2）分类：在缺陷调查阶段，需要进行分类的属性包括缺陷引起的实际原因、缺陷的来源、缺陷的具体类型等。同时，对缺陷识别阶段中的分类信息，根据需要进行检查和更新。

（3）确定影响：根据缺陷调查阶段的分析结果，对缺陷识别阶段的影响分析进行更新。

表 4-5 列举了调查阶段的支持数据。

表 4-5 调查阶段的支持数据表格

接 收 确 认	验 证
接收日期	缺陷的来源
指定的报告号	缺陷识别过程的数据
调查员	
姓名	
代号或职能范围	
电子邮件地址	
电话号码	
计划的调查开始日期	
计划的调查结束日期	
实际的调查开始日期	
实际的调查结束日期	
工时	
接收确认的日期	
调查使用的资料	
名称	
ID	
版本	

3. 改正

根据缺陷调查阶段中得到的结果和信息，就可以采取改正措施解决引起缺陷的错误。采取的行动可能是修复缺陷，也可能是针对开发过程和测试过程的改进建议，以避免在将来的项目中重复出现相似的缺陷。针对每个缺陷的修复，需要进行相关的确认测试和回归测试，避免由于缺陷的修复而影响原有的功能。缺陷改正阶段的主要活动如下。

（1）记录。在缺陷改正阶段，需要记录改正缺陷的相关支持数据信息，包括需要修改的条目、需要修改的模块、修改的描述、修改的负责人、计划修改开始的时间、计划修改完成的时间等。

（2）分类。当合适的修改计划或者活动确定以后，需要对下面的信息进行分类：缺陷修复的优先级（例如，是马上修改、延期修改还是不修改）、缺陷的解决方法（如表 4-6 所示）、缺陷修复的改正措施等。

（3）确定影响。对在缺陷识别阶段、缺陷调查阶段中得到的影响分析进行合适的检查，并在需要的时候进行更新。

表 4-6　缺陷改正的分类表——解决方法

类　　别	符合度要求	代　　号	分　　类
解决方法 AC100	强制性	AC110	即时解决方法
		AC111	修改软件
		AC112	更新项目文档
		AC113	培训操作员
		AC114	修改测试软件
		AC115	外部供应商/第三方
	强制性	AC120	最终解决方案
		AC121	修改软件
		AC122	更新项目文档
		AC123	培训操作员
		AC124	修改测试软件
		AC125	外部供应商/第三方
	强制性	AC130	延期解决方案
		AC131	在以后版本中修改
		AC132	申请放弃
	强制性	AC140	不做修改
		AC141	没有发现问题
		AC142	申请放弃
		AC143	修改不合理
		AC144	修改不可鉴别
		AC145	作废

4．总结

缺陷总结阶段的主要活动如下。

（1）记录：在缺陷总结阶段，需要对一些支持数据信息进行记录，例如，缺陷关闭时间、文档更新完成时间等。

（2）分类：针对缺陷进行确认测试和相关的回归测试以后，就可以将缺陷的状态进行分类，例如，关闭状态、延迟状态或者合并到其他项目中去等。

（3）确定影响：对在缺陷识别阶段、缺陷调查阶段和缺陷改正阶段中得到的影响分析进行合适的检查，并在需要的时候进行更新。

表 4-7 列出了缺陷总结阶段的分类数据。

表 4-7　缺陷总结阶段的分类表

类　　别	符合度要求	代　　号	分　　类
缺陷总结 DP100	强制性	DP110	已关闭
		DP111	已完成缺陷改正
		DP112	不是错误
		DP113	不属于项目范围（不能解决）
		DP114	外部供应商问题
		DP115	重复问题
		DP120	延期改正
		DP130	与其他缺陷合并
		DP140	划归其他项目

5．案例

本章节介绍一个根据 IEEE Std 1044—1993 标准制定的缺陷管理生命周期案例，如图 4-2 所示。

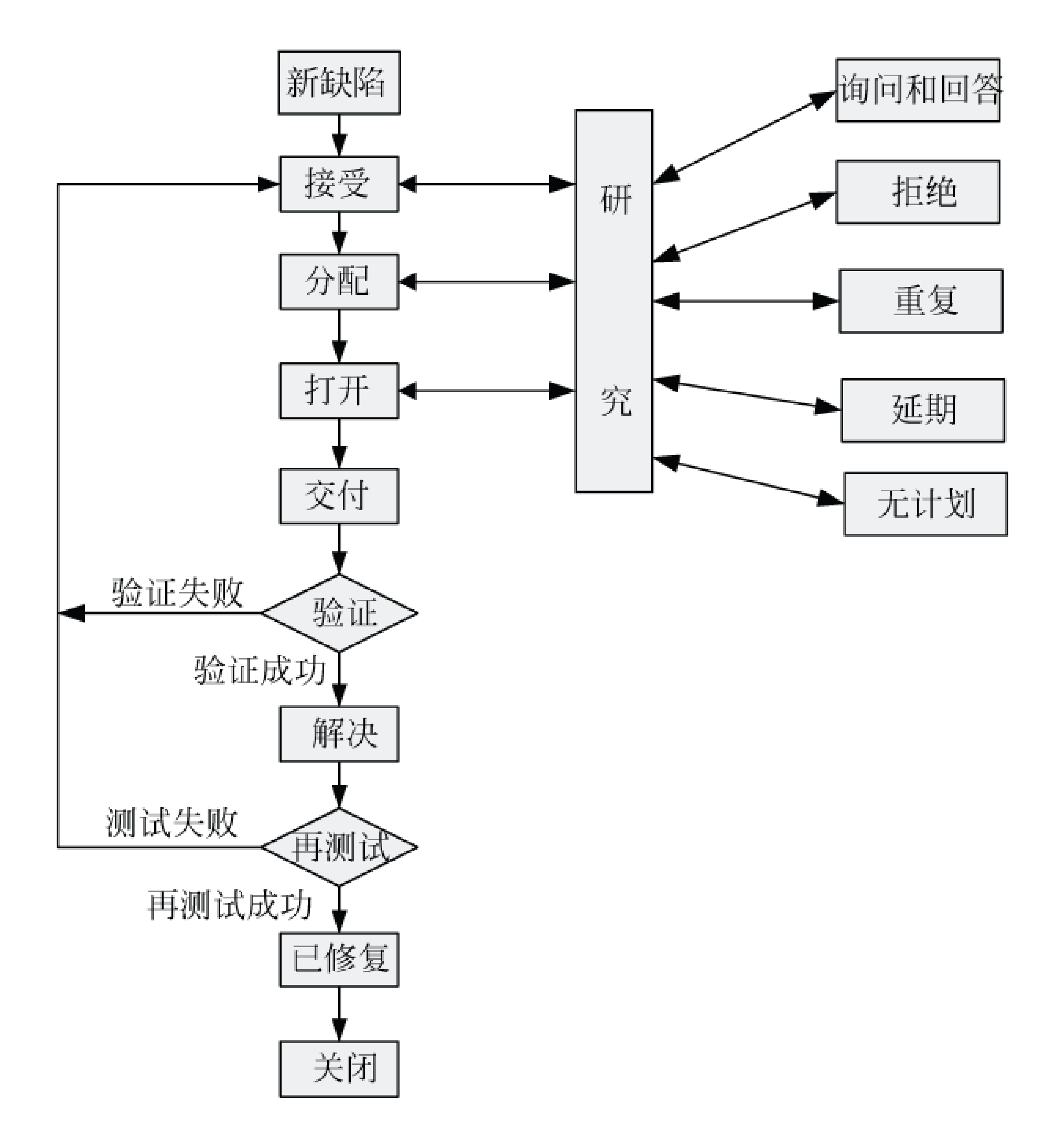


图 4-2　缺陷状态转换图

图 4-2 是某项目的缺陷生命周期中的缺陷状态转换图。下面分别从角色、状态、严重程度和优先级 4 个方面阐述该项目使用的缺陷生命周期。

1）相关角色

（1）测试人员：主要是指发现和报告缺陷的测试人员。通常情况下，测试人员需要对该缺陷后续相关的状态负责，包括回答相关人员对这个缺陷信息的询问，以及在后续版本上进行确认测试和回归测试。

（2）开发人员：主要指对缺陷进行调查和修复的开发人员。需要注意的是，在提交软件版本给测试人员正式再测试之前，需要对修改后的缺陷在开发环境上进行验证。

（3）缺陷评审委员会：主要由项目经理、测试经理、质量经理、开发经理以及资深的开发人员、测试人员等组成。他们对缺陷进行评估和确认，并将确认的缺陷分配给相应的开发人员进行修复，同时对有争议的缺陷进行仲裁。

（4）版本经理：负责将已经解决的缺陷相关的配置信息合并到新的版本。

2）缺陷状态

（1）新缺陷（New）：测试过程中发现的软件产品相关的异常，通常由测试人员提交缺陷报告，当然开发人员自己在组件测试或代码走查过程中发现缺陷时也可以提交缺陷报告，还有可能是从软件产品使用的最终用户或使用现场反馈得到的缺陷报告。发现缺陷的阶段可以是以下几个。

① 需求和设计阶段：文档评审过程中发现的缺陷。

② 编码阶段：代码评审和代码静态分析过程中发现的缺陷。

③ 测试阶段：动态测试过程中发现的缺陷。

④ 使用阶段：用户反馈的缺陷。

（2）接受（Accepted）：相关人员提交的缺陷报告，需要经过缺陷评审委员会的评审。缺陷评审通过后，将缺陷状态置为接受。缺陷评审委员会评审的主要内容如下。

① 报告中描述的问题是否确实是缺陷；

② 提交的缺陷报告是否符合要求。

（3）分配（Assign）：将缺陷状态为接受的缺陷分配给相关人员进行问题定位和修复。相应的缺陷状态被置为分配。

（4）打开（Open）：当缺陷处于打开状态时，说明开发人员已经开始对该缺陷进行修复。

（5）交付（Deliver）：解决缺陷的方法已经找到，并且已经将修改后的代码等打上标签，交付给版本经理。

（6）解决（Resolved）：版本经理将相关的标签等合入某个版本，交付给相关的开发小组进行测试，测试通过，则缺陷状态置为解决。

（7）已修改（Fixed）：版本经理将已经解决的缺陷标签集成到某个软件版本，交付给相关的测试小组进行确认测试，测试通过，则缺陷状态为已修改。

（8）结束（Closed）：缺陷状态处于已修改后，缺陷评审委员会对整个缺陷修复过程进行评审，评审通过后将缺陷状态修改为结束状态。

上面介绍的是在缺陷管理过程中主要涉及的状态，或者是在缺陷处理顺利时所经历的状态。实际上，缺陷还有一些其他的状态。

（1）研究（Investigate）：当缺陷分配给开发人员时，开发人员并不是都能直接找到相关的解决方案。开发人员需要对缺陷和引起缺陷的原因进行调查研究，这时候可以将缺陷状态改为研究状态。

（2）询问和回答（Query&Reply）：假如负责缺陷修复的人员认为缺陷描述的信息不够明确，或希望得到更多与缺陷相关的配置和环境条件等信息，可以将缺陷状态改为询问和回答。

（3）拒绝（Declined）：缺陷评审委员会通过讨论研究，认为提交的问题不是缺陷；或通过开发人员的研究分析，认为不是缺陷，开发人员可以将具体的拒绝理由加入到缺陷描述中，缺陷评审委员会据此将缺陷状态修改为拒绝。

（4）重复（Duplicated）：缺陷评审委员会认为该缺陷和某个已经提交的缺陷描述的是同一个问题，可以将该缺陷状态设置为重复。

（5）延期（Deferred）：缺陷不在当前版本解决。

（6）无计划（Unplanned）：在用户需求中没有要求或计划。

3）严重程度

缺陷的严重程度指的是假如缺陷没有修复，缺陷对软件产品质量的破坏程度，即此缺陷的存在对软件功能特性和非功能特性产生的影响。缺陷的严重程度关注的是缺陷引发的问题对客户的影响程度。因此，确定缺陷严重程度时，应该从软件产品最终用户的角度进

行判断，即根据缺陷会对用户使用造成的影响程度来确定。通常情况下，缺陷对用户造成的影响越大，其缺陷的严重程度越高。下面是缺陷严重程度的分类例子。

（1）严重程度 1（致命的）：软件产品在正常的运行环境下无法给用户提供服务，并且没有其他的工作方式可以补救；或者软件产品存在的失效会造成人身伤害或危及人身安全，例如：

① 软件产品的问题会影响软件产品的数据传输；

② 用户在使用正常操作时，软件产品无法提供正常服务；

③ 软件产品正常运行下会出现运行环境崩溃，造成数据的丢失；

④ 无法提供软件产品的主要功能；

⑤ 软件产品的运行可能会造成人身伤害。

（2）严重程度 2（严重的）：极大地影响软件产品提供给用户的服务，或者严重影响软件产品质量要求或者基本功能的实现，例如：

① 软件产品中的硬件不时会出现重启，但没有影响所提供的传输性能；

② 软件产品正常运行下，会出现软件产品重启或挂起，但不影响软件产品的数据传输；

③ 软件产品提供的操作界面无法操作，或者会产生错误的提示信息。

（3）严重程度 3（一般的）：软件产品功能存在缺陷或功能需要增强，但有相应的补救方法解决这个缺陷，例如：

① 软件产品的硬件部分出现失效，但软件产品没有上报相应的告警；

② 功能特征设计不符合系统的需求，但不影响软件产品提供的业务，并且有相应的补救方法；

③ 本地化软件的某些字符没有翻译或者翻译错误。

（4）严重程度 4（轻微的）：细小的问题，不需要补救方法或对功能进行增强；或者操作不方便，容易使用户误操作，例如：

① 上报的信息不符合软件产品的需求，描述不精确或可能对用户有些误导；

② 图形用户界面问题，显示的信息不精确或可能产生歧义。

4）优先级

优先级是处理软件缺陷的先后顺序的指标。确定缺陷的优先级更多的是站在软件开发和软件测试的角度来进行考虑。确定缺陷的优先级有时候可能并不是纯技术的问题，还需要考虑修复缺陷的难度和存在的风险，因此，确定缺陷优先级是一个复杂的过程。同时，优先级的确定也需要考虑缺陷发生的频率和对目标用户的影响。下面是缺陷优先级的分类例子。

（1）优先级 1（立即）：软件产品中的缺陷会导致用户无法正常开展业务或工作，或运行中的测试无法继续。该缺陷需要立即修复，或采取临时措施（例如，给当前软件版本打补丁的方式）。

（2）优先级 2（下次发布）：在下次常规的软件产品版本发布或下次（内部）测试对象版本交付时实施修正。

（3）优先级 3（必要时）：在受影响的软件产品组件进行修订时再进行修正。

（4）优先级 4（未决）：尚无修正计划。

缺陷的严重程度和优先级是含义不同但相互联系密切的两个概念，它们从不同角度描述了软件缺陷对软件产品质量、用户、开发过程的影响程度和处理方式。一般来说，严重程度高的缺陷具有较高的优先级。严重程度高说明缺陷对软件造成的质量危害性大，需要优先处理，而严重性低的缺陷可能只是软件的瑕疵，可以稍后处理。但是优先级和严重程度并不总是一一对应的，也存在优先级低但严重程度高的缺陷，或者优先级高但严重程度低的软件缺陷。修改软件缺陷并不是纯技术的问题，有时候需要考虑软件版本发布和质量风险等因素。下面是关于缺陷严重程度和优先级设置方面的一些建议。

（1）如果某个严重的缺陷只在非常极端的条件下产生，则可以将缺陷的优先级设置得比较低；

（2）如果修正一个软件缺陷需要重新修改软件产品的实现整体架构，可能会产生更多的潜在缺陷，而且市场要求尽快发布软件版本，那么即使这个缺陷严重程度很高，也需要仔细考虑是否需要修改；

（3）对于有些缺陷，可能它的严重程度很低，例如，界面单词拼写错误，但假如这是公司的名称或者商标，则这个缺陷的优先级就很高，必须尽快进行修复，因为这个关系到软件系统和公司在市场上的形象。

缺陷管理过程由不同的缺陷状态组成，不同利益干系人关注的缺陷状态和职责是不同的。对于测试人员而言，下面的三类缺陷状态是特别重要的，需要测试团队采取相应的行动。

1）初始状态

（1）通常是测试人员发现缺陷并提交缺陷报告，以帮助开发人员收集信息和重现该缺陷。详细内容请参考 4.3 节。

（2）“初始”状态也可以称为“打开”状态，或者“新的”状态；图 4-2 中指的是“新缺陷”状态。

2）退回状态

（1）该状态指的是缺陷报告接收人拒绝接收缺陷报告，或者需要测试人员提供更进一步的信息.

（2）假如缺陷处于退回状态，说明缺陷报告中提供的初始信息或者测试本身存在不足。在这个状态测试人员必须提供补充信息，或确认拒绝该缺陷报告。测试经理应该监控此类缺陷并分析其中的原因，以免出现过高的缺陷退回率。

（3）“退回”状态也可能被称为“拒绝”状态或“澄清”状态。图 4-2 中指的是“询问和回答”状态、“拒绝”状态、“重复”状态、“延期”状态和“无计划”状态。

3）确认测试状态

（1）当缺陷处于该状态时，测试人员应该按照缺陷报告中重现失效的步骤，运行确认测试以判断开发人员是否正确地修复了缺陷。如果确认测试表明缺陷已被修复，测试人员应关闭缺陷报告。如果确认测试显示缺陷没有被修复，测试人员应该再次打开缺陷报告，让之前安排的缺陷负责人继续修复缺陷。

（2）“确认测试”状态也可能被称为“解决”状态或“验证”状态。图 4-2 中指的是“再测试”状态。

4.2.2 无效和重复缺陷的管理

测试人员在测试过程中发现测试实际结果与期望结果不一致，即观察到异常情况，必须要做进一步的调查。有些情况下，出现异常并不是由于软件产品中的缺陷导致的，而是由于测试环境、测试数据、测试件的其他原因引起的，或者是测试人员自己的误解而导致的。即测试人员在测试过程中会遗漏缺陷或者误报缺陷，包括：

（1）假阴性结果（假通过结果/漏测缺陷）：测试人员未能识别出测试对象中的缺陷。

（2）假阳性结果（假失败结果/误报缺陷）：测试人员报告了测试对象中实际不存在的缺陷。

假阴性结果（漏测缺陷）主要是由于测试人员能力方面的限制而导致，有时也由于测试资源等方面的限制而没有发现缺陷，例如，测试团队缺乏安全性测试工具，导致无法发现安全漏洞问题。而假阳性结果主要是由于测试人员对测试对象的理解方面存在不足而导致的，这类缺陷报告一般会被取消或者作为无效缺陷报告而被关闭。

另外，测试人员在测试过程中会提交一些重复的缺陷报告，甚至在有些情况下，两个缺陷的外部表现症状是完全不相关的，而开发定位之后发现是由相同的根本原因引起的。此时开发团队只会保留其中的一个缺陷报告，将其他重复的缺陷报告置于关闭状态。

尽管无效缺陷报告和重复缺陷报告会影响团队的效率，但是一定数量的此类缺陷报告是不可避免的，测试经理应该接受这种情况。测试经理需要做的是如何实现无效和重复缺陷报告与高效测试之间的平衡。假如测试经理试图消除所有的无效和重复缺陷报告，必定会导致假阴性结果的数量增加，因为测试人员会担心出错而不敢提交一些缺陷报告，这又会降低测试团队发现缺陷的有效性。

1．无效缺陷报告

在报告缺陷之前得到开发人员的确认，是一种比较有效的避免无效缺陷报告的手段。即测试人员在测试环境中发现异常时，假如条件允许，提交缺陷报告前可以和相关开发人员进行确认。当然，对于非常容易判断的功能性缺陷或者其他非常明确的缺陷类型，测试人员可以直接提交缺陷报告。而对于稳定性、功能行为表现复杂或者难以复现的缺陷，可以要求开发人员在测试现场确认发现的缺陷，这样可以提高项目的整体效率，有利于团队之间的合作。主要表现在以下几个方面

（1）首先，开发人员可以现场查看被测对象的输出信息和异常表现，有利于开发人员后续的缺陷复现和定位解决。但是需要注意的是：开发人员应该避免直接在测试环境中进行缺陷的定位和修改，特别是在时间和资源占用比较多的情况下。

（2）其次，开发人员可以帮助测试人员确认发现的问题是不是缺陷，避免测试人员提交一些不是缺陷的缺陷报告（例如，由于测试人员理解错误、配置错误等而导致的一些系统异常，即假阳性结果），同时避免开发人员在不是缺陷的缺陷报告上面浪费时间和精力，提高测试团队和开发团队的效率。

（3）第三，有利于开发人员和测试人员的沟通，包括对软件产品行为的理解，从而在项目团队之间更好地合作；

2. 重复的缺陷报告

缺陷信息共享是减少提交重复缺陷报告的一种手段，指的是测试人员在测试执行过程中发现的缺陷，在不同项目成员之间进行共享，以减少重复缺陷报告的提交。低级别的测试（例如，组件测试或集成测试）通常由开发人员完成；而高级别的测试（例如，系统测试或验收测试）由独立的测试团队完成。在开发人员正式提交软件版本的系统测试前，开发人员需要进行基本功能的测试，并提供关于该版本的发布说明。由于在每个测试级别和不同的测试阶段都有可能发现缺陷，因此，在不同的项目成员之间进行缺陷信息的共享显得尤为重要。主要包括：

（1）开发人员和测试人员之间的缺陷信息共享，例如，在开发人员提供的版本发布说明中，提供当前版本中存在的主要缺陷、对后续测试的影响以及建议的测试重点等。这些信息可以为测试人员进行的测试提供参考，同时可以避免测试人员提交一些已经存在的缺陷报告，从而避免浪费时间和精力，不断提高测试的有效性和效率。

（2）测试人员之间的共享。在测试执行过程中，测试人员应该将提交的缺陷信息在项目的测试团队中进行共享，例如，通过邮件的形式，或者通过测试团队内部简短的会议等。特别是严重的缺陷和可能会导致阻塞很多测试用例执行的缺陷，应该在测试团队内部发布该缺陷信息。通过在测试人员之间共享缺陷信息，可以在团队内部形成共同分析和解决问题的氛围，提高团队的测试热情和测试效率。

4.2.3 跨职能缺陷管理

尽管测试团队是缺陷管理过程的积极参与者，但一般会有跨职能缺陷管理团队，负责整体的缺陷管理过程。除了测试经理，跨职能缺陷管理团队通常还包括开发经理、项目经理、产品经理，以及其他软件项目的主要利益干系人。

跨职能缺陷管理团队的成立，其主要目的是更加高效地在不同团队之间进行缺陷沟通和协调，从而实现更好的缺陷跟踪和管理，特别是可以避免不同团队之间对缺陷的不同理解而导致的冲突。

测试人员提交缺陷报告后，经常会出现测试人员认为这是软件系统的一个缺陷，而开发人员认为这是正常的系统功能。到底是缺陷还是正常功能？这是测试人员和开发人员在缺陷问题上经常出现的分歧。

对于测试中发现的功能问题，开发人员和测试人员之间比较容易达成共识。但是，对于软件非功能相关的问题，由于相关的规格说明中经常没有进行详细定义，因此容易出现缺陷认定方面的争议。此时，测试人员需要具有良好的沟通能力，并对测试理论和领域知识有比较深入的了解。下面是在缺陷认定出现不一致的时候，缺陷相关人员可以借鉴的一些建议。

（1）测试人员应该让开发团队理解提交缺陷的目的并不是对开发人员的成果进行挑刺，相反，测试人员和开发人员的目标是一致的，都是为了提高软件产品的质量，满足客户的需求。

（2）需要和开发人员进行有效的沟通，让开发人员理解测试的目的不只是针对软件的功能而开展，还包括可靠性、易用性、效率、维护性、可移植性等其他质量特性。因此，

发现非功能特性的缺陷时，开发人员不应该因为系统需求规格说明中没有详细定义，而否认存在的问题。

（3）测试人员应该站在用户的角度对软件的使用质量特性进行有效的测试，包括软件的有效性、生产率、安全性以及满意度等。

另外，除了不同团队之间进行有效沟通之外，跨职能缺陷管理团队在这个问题上也可以发挥作用。假如测试团队发现的缺陷，没有合理的测试依据，例如需求规格说明作为参考，那么跨职能缺陷管理团队可以最终讨论决定提交的缺陷是不是问题。

跨职能缺陷管理团队还可以确定缺陷是否在当前软件版本中进行修复？假如不是，要在计划到哪个版本进行修复等问题上做出明确的决定，以及修复的优先级等，而不是把这些问题让开发团队或者测试团队之间争论确定。跨职能缺陷管理团队在做这些决定时，必须考虑成本、风险和收益等的平衡。

需要注意的是，缺陷管理工具不能替代团队之间的沟通与交流，跨职能缺陷管理团队也不能。沟通交流、充分的工具支持、定义明确的缺陷管理生命周期和缺陷管理委员会的充分参与对有效和高效的缺陷管理来说，都是不可或缺的。

4.3 缺陷报告信息

软件开发过程中通过静态测试识别缺陷，或者通过动态测试发现失效时，相关人员就需要收集相应的信息和数据，并以缺陷报告的形式提交到缺陷管理系统。良好的缺陷报告信息，不仅可以帮助开发人员定位和修复缺陷，同时可以有效评估软件产品质量和测试进度，并为评估和改进过程能力提供度量。

根据 IEEE Std 1044—1993 中的定义，缺陷生命周期由缺陷识别、缺陷调查、缺陷改正和缺陷总结 4 个阶段组成。对于每个阶段，分别由记录、分类和确定影响三个活动组成，这三个活动的对象就可以构成缺陷（报告）的组成要素。不同的组织或项目，可以将 IEEE Std 1044—1993 中定义的要素映射到相应的缺陷报告模板中，并且根据需要将缺陷相关的要素名称进行修改。同时，也可以根据需要对缺陷报告的组成内容进行裁减。下面是缺陷报告模板的例子，它的主要组成如下。

（1）缺陷 ID：唯一标识缺陷的标识符，缺陷 ID 在整个缺陷管理系统中应该是唯一的。

（2）所属产品：缺陷所属的产品或项目，例如：iBAS R1.0。

（3）摘要：对缺陷进行简短而清晰的描述，使缺陷相关人员可以快速了解缺陷的内容。

（4）测试环境：包括发现缺陷的测试平台、操作系统、测试软件版本、系统配置等。

（5）日期和时间：缺陷发现的日期和时间。

（6）提交者：缺陷报告的提交者，可以是开发人员、测试人员、用户等。同时，这里也可以提供缺陷提交者的电子邮件地址，这样缺陷管理工具可以在缺陷状态发生变更的时候，直接将状态信息发送给缺陷提交人。

（7）优先级：可以是缺陷立即修改、下次发布修改、必要时再修改、未决修改等。

（8）严重程度：缺陷可以是致命的、严重的、一般的或轻微的等。

（9）发现阶段：缺陷在软件开发生命周期的哪个阶段被发现，可以是需求阶段、设计

阶段、编码阶段、测试阶段、生产阶段、用户反馈等。

（10）缺陷复现步骤：包括发现缺陷的输入、测试步骤、期望的输出结果、实际的输出结果以及存在的异常情况等。

（11）缺陷其他信息：发现缺陷的测试用例编号，和缺陷相关的一些日志文件、告警信息和输出信息等。

☆示例：iBAS R1.0 中针对 IGMP 功能提交的一个缺陷报告

===基本信息:

缺陷 ID：SHms07496	状态："验证"状态
类型：缺陷	提交时间：20081110
发现项目：iBAS R1.0	所属模块：IGMP
缺陷提交者：XXX	缺陷提交者 E-mail：XXX@ABC.com.cn
严重程度：1	优先级：1

===缺陷信息

摘要：IGMP：删除 IGMP 组播源导致系统重启。

发现阶段：系统测试

缺陷引起原因：代码编写错误

缺陷处理人：YYY

是否可以重现：是

===缺陷描述

测试环境：系统测试网络拓扑 1（见系统测试用例规格说明）。

测试步骤:

（1）打开系统 IGMP 功能。

（2）增加一个 IGMP 组播源，组播源地址为 224.0.0.5。

（3）删除步骤（2）中配置的组播源。

期望结果:

（1）系统功能打开成功。

（2）IGMP 组播源增加成功。

（3）能够成功删除步骤（2）中增加的组播源。

实际结果:

步骤（1）、步骤（2）的实际结果和期望结果一致。

步骤（3）导致软件系统出现非正常重启。系统重启后该组播源并未被删除。

系统出错信息：见附件。

系统调试信息：见附件。

软件开发生命周期的每个阶段都会发现缺陷，缺陷发现的越早，提交缺陷报告和跟踪软件项目状态所需的信息和数据相对也越少（例如，需求评审阶段和单元测试阶段）。因此，收集缺陷管理和跟踪所需的信息应该根据缺陷发现阶段的不同进行适当的调整。然而，为了能对软件开发生命周期内不同阶段之间的缺陷数据进行有意义的比较，必须将不同阶段发现缺陷的关键信息保持一致。

收集和分析每个阶段发现的缺陷数据，有助于监控测试进度和评估出口准则。例如，缺陷信息可以支持缺陷密度分析、缺陷发现和解决的趋势分析，从发现缺陷到缺陷解决的平均时间和失效强度(如平均故障间隔时间分析 MTBF)。涉及缺陷管理的各类标准和文档，例如，ISO/IEC 9126，IEEE 829，IEEE 1044 和正交缺陷分类 ODC 等，可以作为测试经理选择和收集缺陷报告所需信息的重要参考依据。

无论最终决定收集哪些缺陷相关的信息，测试人员在提交缺陷报告时，必须确保输入信息的完整、简明、准确、客观、相关和及时，这一点很关键。尽管测试人员与其他利益干系人之间面对面的沟通交流，可以有效避免一些不符合要求的缺陷报告带来的资源浪费等问题，但缺陷报告中缺少某些字段信息，还是会对软件项目状态、测试进度和过程能力评估带来影响。

4.4 使用缺陷报告信息评估过程能力

测试过程中提交的缺陷报告和报告中提供的信息，对软件项目的监控和汇报非常重要。尽管度量对过程的影响主要是在 ISTQB 专家级测试管理大纲中讨论（例如，使用度量数据评估测试过程效率和有效性），但是作为高级大纲中测试经理的角色，应该意识到缺陷报告和相关信息对评估测试能力和软件开发过程能力的意义。主要表现在以下几个方面。

（1）根据缺陷引入阶段、识别阶段和移除/修复阶段的信息，评估阶段缺陷密度，并提出改善各阶段缺陷发现效率的方案；

（2）根据缺陷引入阶段的信息，对引入缺陷最多的阶段进行帕累托图（Pareto）分析，使改进更具针对性，减少缺陷总数；

（3）根据缺陷根本原因信息确定缺陷引入的潜在原因，支持过程改进，以减少缺陷总数；

（4）根据缺陷引入、识别和移除/修改的阶段的信息进行质量成本分析，尽量降低与缺陷相关的成本；

（5）根据缺陷集群效应，找出缺陷比较集中的组件，以便更好地理解技术风险（对基于风险的测试来说）和考虑重新设计问题较多的组件。

☆示例：缺陷引入阶段–发现阶段评估矩阵

通常在缺陷报告中会有两个字段：缺陷引入阶段、缺陷发现阶段，而引入和发现阶段可以是软件开发生命周期的任何阶段。测试经理可以根据引入阶段和发现阶段绘制出一个矩阵（简称缺陷引入阶段 – 发现阶段评估矩阵），不仅可以评估缺陷阶段遏制能力，也可以分析软件开发生命周期中各个阶段工作产品的质量，并找到最需要改进的环节。

表 4-8 是缺陷引入阶段 – 发现阶段评估矩阵的例子。其中，矩阵的每行表示该阶段发现的各个阶段所产生的缺陷数；而矩阵的每列表示该阶段引入的缺陷遗漏到后续阶段的缺陷数。例如，需求阶段发现的需求缺陷是 5 个；设计阶段发现了需求阶段引入的缺陷 20 个，以及设计阶段引入的缺陷 30 个。阶段缺陷移除率的定义：

阶段缺陷移除率=（本阶段发现的缺陷数/本阶段引入的缺陷数）× 100%

例如，需求阶段的缺陷移除率=5/92=5.43%。其中需求阶段发现的缺陷数目是 5；而需求阶段引入的缺陷数目=5+20+15+1+50+1=92 个。同样，根据缺陷移除率，就可以得到缺陷遗漏率=1–缺陷移除率。缺陷移除率和缺陷遗漏率两个指标，可以反映该阶段的缺陷遏制能力，以及本阶段质量控制措施的有效性。

表 4-8　缺陷引入阶-发现阶段评估矩阵

发现阶段	引入阶段			
	需求缺陷	设计缺陷	编码缺陷	缺陷总计
需求阶段	5			5
设计阶段	20	30		50
编码阶段	15	10	45	70
集成测试	1	2	5	8
系统测试	50	45	150	245
验收测试	0	1	5	6
用户反馈	1	3	16	20
缺陷总计	92	91	221	404
缺陷移除率	5.43%	32.97%	20.36%	

以需求缺陷为例，根据每个阶段发现缺陷的数目分析过程的有效性。需求阶段引入的缺陷（即需求缺陷），大部分是通过系统测试发现的，而在前期，特别是需求阶段的缺陷移除率非常低（仅 5.43%），即需求阶段的缺陷遗漏率很高。

缺陷越到后期发现，用于缺陷复现、定位和移除/修复的时间和成本就会越高。测试经理必须分析为什么在前期发现的缺陷这么少，找到其中的原因，例如：

（1）需求阶段的需求评审是否投入的工作量不足？

（2）软件项目的管理层是否对需求评审支持不足？

（3）参与需求评审的利益干系人是否在评审技能方面不足？例如：经验不足。

（4）是否没有合适的评审工具的支持？例如：评审检查表。

通过分析测试经理后续就可以采取合适的整改措施，不断提升需求评审的效率和有效性，从而改进需求阶段的缺陷阶段遏制能力。

☆示例：结束

为了有效地评估和改进过程能力，测试经理必须在测试分析师和技术测试分析师的支持下，收集和分析所需的缺陷度量数据。而收集和分析缺陷数据必定是需要考虑成本的，因此测试经理需要在时间、成本、质量等方面进行平衡。测试团队有时候会认为跟踪和分析开发过程中发现的缺陷，不仅浪费费用且降低了开发进度，因此对度量数据的收集和分析带有排斥的心态。但事实上这种做法会极大降低测试过程和开发过程的透明度，不仅很难对过程进行监控，同时由于缺乏可靠的数据，难以实施过程改进。

小结

尽早发现缺陷是测试的重要目的之一。为了保证有效而及时地解决测试过程中发现的缺陷，组织内需要建立一套完整的过程和规则对缺陷进行跟踪和管理。需要从缺陷的发现、缺陷提交、缺陷分类、缺陷修复、解决方案验证等整个过程进行跟踪。

测试应该贯穿于整个软件开发生命周期，因此，缺陷可以在软件开发生命周期的任何一个阶段被发现和修复，例如，需求阶段、设计阶段、测试阶段等。对缺陷的跟踪和管理是通过不同角色和职责的人员参与，推动缺陷的一系列状态演变的过程。

缺陷报告是开发人员进行缺陷定位和修复的基础，因此，良好的缺陷报告的构成非常重要，它应该由一些基本的元素组成。不同的组织和项目，可以对 IEEE Std 1044—1993 中定义的要素进行合理的裁减，得到适用于自身特点的缺陷报告模板。

测试过程中发现的缺陷数目、缺陷引入阶段、发现缺陷阶段、缺陷所在功能模块等缺陷数据，是评估和改进过程能力的基础。具体需要收集的缺陷相关数据，依赖于项目团队的管理要求和目的，定义合适的缺陷报告所需的属性字段，从而更好地服务于基于缺陷信息的过程监控和过程改进。

模拟题

36．CTAL-ATM_LO-4.2.1

TM-4.2.1(K3) 为测试组织开发缺陷管理过程，包括缺陷报告流程，用于在测试生命周期中监控项目缺陷。

问题：

图 4-3 显示了一个不完整的缺陷管理过程，三个状态（状态 X、Y 和 Z）必须进行合适的命名。

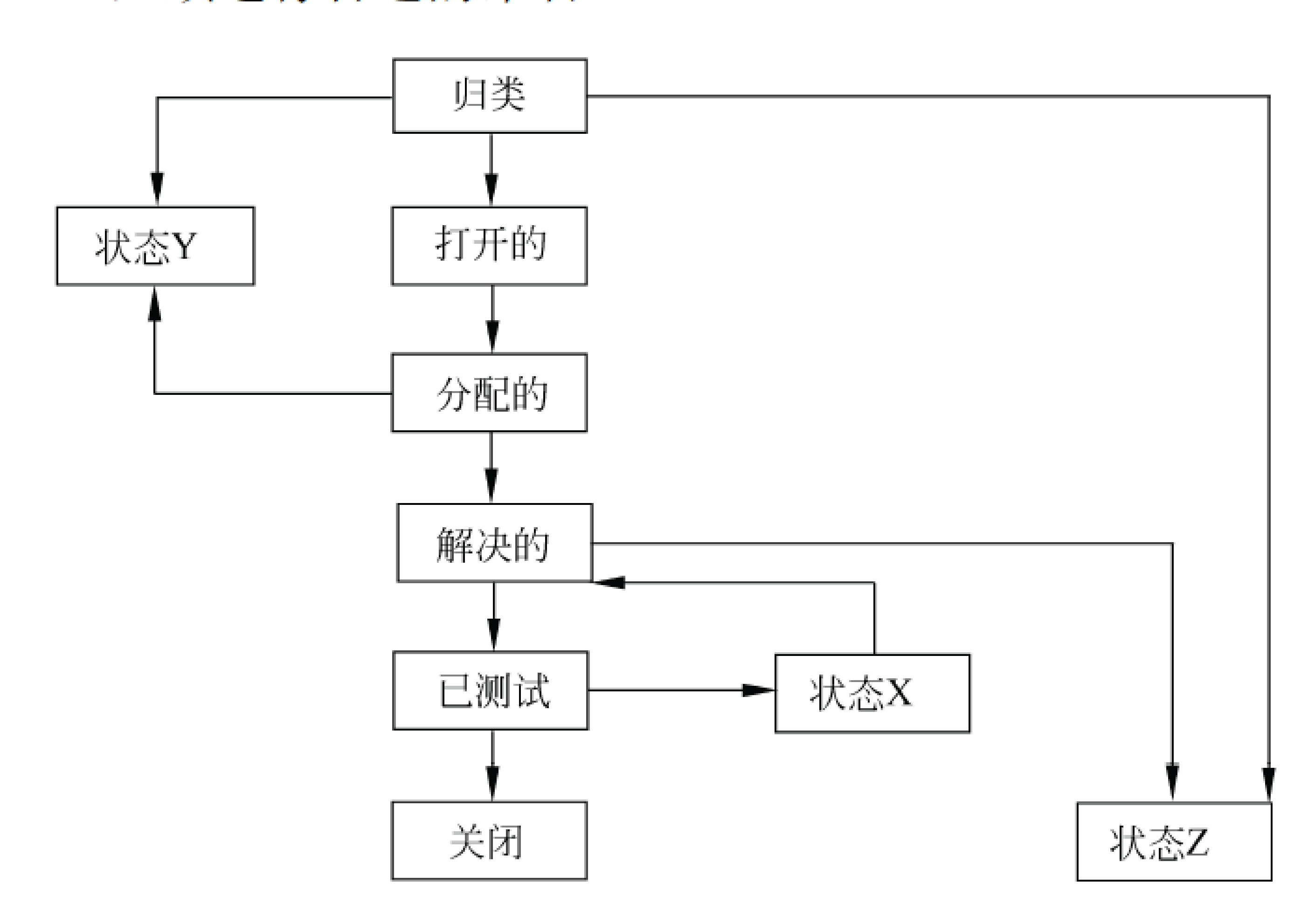


图 4-3　不完整的缺陷管理过程

下面哪个选项的内容可以正确地完成该过程？

选项：

A．状态 X – 再打开；状态 Y – 拒绝；状态 Z – 延期
B．状态 X – 再测试；状态 Y – 新的；状态 Z –阻塞
C．状态 X– 重复；状态 Y – 未确认；状态 Z – 中止
D．状态 X– 已验证；状态 Y – 评审；状态 Z – 已修复

解释：

A．正确。起始状态和分配状态都可能拒绝缺陷（因此转换为拒绝状态）。假如测试发现缺陷没有被修复，则将它置为再打开状态。起始状态和分配状态都可能决定该缺陷延期（可能延到下个版本）。

B. 不正确。测试之后马上又再测试没有意义。新缺陷与打开的缺陷往往是同义词。假如状态 Z 是阻塞，无法从这个状态退出。
C. 不正确。重复的缺陷不需要再分配。状态 Y 不能是未确认状态，因为无法离开这个状态。状态 Z 在某些情况属于中止状态。
D. 不正确。已验证和已测试通常是同义词——因此都不需要。状态 Y 不能是评审状态，因为无法从这个状态退出。 状态 Z 不能是已修复状态，因为已修复与已解决是同义词，无法从状态 Z 直接到关闭状态。

分值：2 分

37．CTAL-ATM_LO-4.2.2

TM-4.2.2(K2) 说明有效的缺陷管理必需的过程和参与者。

问题：

下面哪个选项可以作为缺陷报告状态达到终止状态的正常顺序？假设“正在处理”指的是开发人员或者其他项目干系人正在研究缺陷的一个或者多个状态。

选项：

A. 初始状态、正在处理、返回、取消
B. 初始状态、正在处理、再测试、关闭、延期
C. 初始状态、正在处理、返回、正在处理、再测试
D. 正在处理、初始状态、再测试、关闭

解释：

A. 正确。它符合实际的缺陷管理流程。
B. 不正确。将已经修复的缺陷置为延期状态没有意义。
C. 不正确。再测试不是一个终止状态。
D. 不正确。在缺陷报告之前，不可能处于正在处理状态。

分值：1 分

38．CTAL-ATM_LO-4.3.1

TM-4.3.1(K3)定义在缺陷管理过程中应该收集的数据和分类信息。

问题：

你是某项目的测试经理，该项目的系统测试在第三方提供的软件平台上开展。你收到来自第三方的抱怨：系统测试提供的缺陷信息的完整性无法接受。

下面列出了提交给第三方的缺陷报告中可能遗失的信息条目。

从你的角度，哪三项是最应该加到缺陷报告中的？

选项：

A．修复缺陷的优先级
B．发现缺陷的测试环境
C．复现缺陷的步骤，包括实际结果和期望结果
D．发现缺陷的项目活动
E．缺陷的技术类型
F．缺陷引入、检测和移除的生命周期阶段
G．发现缺陷所在的子系统或者组件

解释：

A．正确。第三方需要这些信息以帮助他们确定优先级。
B．正确。测试环境信息可以帮助他们确定缺陷所需的环境。
C．正确。复现的步骤（和实际结果）有助于他们理解缺陷，而期望结果可以确认测试人员是否理解他们的期望。
D．不正确。第三方已经知道该缺陷报告来自动态系统测试。
E．不正确。当前这个信息没有什么用。
F．不正确。缺陷的检测阶段已经知道（系统测试），缺陷移除阶段还未知（希望是当前阶段）。
G．不正确。定位系统缺陷的位置不属于测试人员的职责范围。
分值。
分值：2 分

39．CTAL-ATM_LO-4.4.1

TM-4.4.1(K2)解释缺陷报告统计信息怎样用于评估测试和软件开发过程的过程能力。

问题：

你的组织已经决定，测试和开发过程改进的第一步是减少在开发阶段引入的缺陷数目。

为了达到这个目标，下面哪个缺陷报告统计信息是最有用的？

选项：

A．缺陷根本原因信息
B．缺陷引入、检测和移除的生命周期阶段
C．缺陷组件信息
D．缺陷移除效率信息

解释：

A. 正确。这可以显示当前的缺陷是从哪里引入的，因此可以关注这些活动以避免将来再次引入缺陷。

B. 不正确。缺陷引入阶段的信息是有用的，但是缺陷检测和移除阶段信息对减少缺陷引入是没有用的。

C. 不正确。这可以作为缺陷集群效应，目标组件需要额外的测试——但无法直接帮助预防缺陷。

D. 不正确。这是移除缺陷的效率，对减少缺陷的引入没有帮助。

分值：1 分

第5章 改进测试过程

本章学习目标如表 5-1 所示。

表 5-1 学习目标

编　号	学习目标描述	级　别
TM-5.2.1	举例说明为什么测试过程改进是重要的	K2
TM-5.3.1	使用 IDEAL 模型定义测试过程改进计划	K3
TM-5.4.1	总结 TMMi 测试过程改进模型的背景，范围和目标	K2
TM-5.5.1	总结 TPI-Next 测试过程改进模型的背景，范围和目标	K2
TM-5.6.1	总结 CTP 测试过程改进模型的背景，范围和目标	K2
TM-5.7.1	总结 STEP 测试过程改进模型的背景，范围和目标	K2

本章相关术语如表 5-2 所示。

表 5-2 术语

英　文	中　文	说　明
Capability Maturity Integration（CMMI）	能力成熟度模型集成	描述有效的产品开发和维护过程的关键元素框架，能力成熟度模型集成包含产品开发、维护计划、工程和管理等方面的最佳实践，是 CMM 的指定的继承版本。[CMMI] 参见 Capability Maturity Model (CMM)
Critical Testing Process（CTP）	关键测试过程	用于测试过程改进的基于内容的模型，大约包括 12 个关键过程。同行和管理者通过其中高度可视的过程对绩效和影响企业收益和声誉的关键任务过程中的表现进行判断
Systematic Test and Evaluation Process（STEP）	系统性测试和评价过程	一种结构化测试方法论，也作为基于内容的模型用于改进测试过程，系统化测试和评估过程（STEP）不要求改进按照特定的次序。参见 content-based model
Test Maturity Model integration（TMMi）	测试成熟度模型集成	与能力成熟度模型集成（CMMI）相关的 5 层测试过程改进框架，描述了有效测试过程的关键因素
TPI-Next		一个用于测试过程改进的业务驱动框架，它具有连续性，描述了有效和高效测试过程的关键要素

5.1 简介

组织内建立测试过程之后，需要在测试过程中不断地收集和分析度量数据和信息，从而实现过程的持续改进。测试分析师和技术测试分析师作为过程改进活动的重要参与者，他们在数据和信息收集与分析方面的支持不可或缺。

本章首先讨论通用的过程改进的过程，然后介绍可用于测试过程改进的各种模型。测试经理是测试过程改进的重要推动力量，熟悉本章的过程改进过程和过程改进模型的内容，有助于测试经理更好地开展过程改进活动。更多测试过程改进内容，可以参考 ISTQB 专家级测试过程改进大纲。

5.2 测试改进过程

软件测试是改进软件产品质量的重要手段，而过程改进技术与方法则可以用来改进过程（例如，开发过程、测试过程等）和由过程生成的工作产品（例如，需求规格说明、测试用例规格说明等）。过程改进技术与方法通过提供改进指南和改进区域，不断改进过程，从而改进过程的交付物。尽管软件测试成本占了软件项目总成本的很大比例，然而在众多的软件过程改进模型中，只有极少部分关注于测试过程改进，例如 CMMi。但是过程改进不仅适用于开发过程，也适用于测试过程。

随着测试重要性的不断提升，测试改进模型也在不断得到发展。常见的测试改进模型包括测试成熟度模型集成（TMMi）、系统化测试和评价过程（STEP）、核心测试过程（CTP）和 TPI-Next 等。这些改进模型解决了软件过程改进模型对测试关注不够的问题。TPI-Next 和 TMMi 提供了跨组织级别的度量，并可作为基准进行参照比较。测试经理需要对所有能用的模型进行研究，以决定最适合组织和项目的模型。

本章节的目的是阐述不同测试改进模型的主要内容和评估过程，而不是推荐它们。

5.2.1 过程改进的介绍

过程改进与软件开发过程和测试过程有着密切的联系。吸取以前的经验教训有助于组织的过程改进，而软件开发和测试也能在这个过程当中受益。戴明改进循环 PDCA（Deming Improvement Cycle）：计划（Plan）、实施（Do）、检查（Check）和行动（Act），已经在业界使用了多年。目前，戴明改进循环 PDCA 仍旧是进行过程改进的首选。

戴明改进循环是由美国质量统计控制之父 Shewhat（休哈特）提出的 PDS（Plan Do See）演化而来的，由美国质量管理专家戴明改进成为 PDCA 模式，所以又称为“戴明环”。戴明改进循环 PDCA 的 4 个过程不是运行一次就结束，而是需要周而复始地进行，一个循环完成后，可以解决一些问题，而未解决的问题则进入下一个循环，螺旋式上升。戴明改进循环是有效进行任何一项工作的合乎逻辑的工作程序，因此，有人称其为质量管理的基本方法。

过程模型可作为改进的起点，通过使用过程改进模型对组织的过程成熟度进行度量。根据评估的结果，对组织过程进行改进。过程评估明确了组织的过程能力，并推动过程改进。同样，过程改进的效果和有效性需要通过过程评估进行度量。

5.2.2 过程改进的类型

测试改进的过程有两种模型：过程参照模型和内容参照模型。

（1）过程参照模型是在进行评估时所使用的指导框架，通过与模型比较以评价组织的能力，并在此框架下评价组织；

（2）内容参照模型用于在评估完成之后进行过程改进。

有些模型同时涵盖了上述两种模型的特点。测试实践中有时不需要过程改进模型，测试团队也可以通过问题分析法或者回顾会议等手段实施测试过程改进。

5.3 改进测试过程

为了达到更高的成熟度和更专业的水平，IT 业已经开始参照测试过程改进模型开展改进工作。行业标准模型有助于开发跨组织的度量与分析（可用于比较）。阶段式模型（例如 TMMi 和 CMMi）提供的比较标准供不同的公司和组织使用；持续性模型允许组织拥有更多的自由度，根据所确定的优先级实施过程改进。针对测试领域的过程改进需要，出现了多个过程改进的标准和模型，包括 STEP、TMMi、TPI-Next 和 CTP 等，这些标准和模型将会在以下各节进行阐述和讨论。

对于各种测试过程评价模型，不同的组织应根据当前的测试过程确定自身的能力成熟度。TMMi 和 TPI-Next 为测试过程的改进提供了明确的操作步骤。STEP 和 CTP 则是向组织提供方法以判定哪些过程改进能带来最大的回报，并让组织选择合适的路径。

测试过程改进实施可以借鉴 IDEAL^{SM} 模型（IDEAL 字母分别代表 5 个步骤的首字母）中定义的 5 个步骤：启动改进过程（Initiating）；诊断目前的情况（Diagnosing）；制订测试过程改进计划（Establishing）；实施改进（Acting）；改进经验教训总结（Learning）。

1．启动改进过程

在过程改进启动之前，首先需要利益干系人对过程改进的目标、目的、范围和评估方法等内容达成一致，并选择过程改进参照的模型。例如，可以选择业界公认的过程改进模型（如 CTP、STEP、TMMi 和 TPI-Next），或者内部自己开发的模型。另外，还要定义在整个改进活动开展期间判断改进成功与否的标准，以及定义度量方法。

2．诊断目前的情况

根据利益干系人已经达成的一致评估方法，识别当前测试过程中存在的主要问题和可以改进的点，并编写测试改进评估报告，内容包括对当前测试过程实践的评估和可能的改进点清单。

3．制订测试过程改进计划

根据前面的测试改进评估报告，对过程改进点清单中的改进点设置相关优先级。设置过程改进点的优先级需要综合考虑投资回报率、风险、组织战略目标、量化的度量指标或者质量收益。设定好改进点优先级之后，需要制订和部署过程改进的计划。

4．实施过程改进

根据测试过程改进计划改进点优先级、时间、资源、角色和职责等定义，实施过程改进，主要包括培训、辅导、过程试运行和最终的全面部署。在实施过程改进过程中，可能会有新的过程改进需求增加进来，也可能对一些原来的改进点进行变更。因此过程改进计划需要随着改进的不断推进，进行相应的调整。

另外，实施测试过程改进过程中，还有个重要的活动是需要将过程改进活动与计划中定义的成功准则进行比较。根据比较结果，调整实施过程改进的活动，以尽快满足测试过程改进的目标。

5．经验教训总结

成功部署测试过程改进之后，需要验证改进带来的收益，包括改进计划中既定的收益

和潜在的其他收益。另外，也需要检查哪些过程改进的成功准则已经满足，哪些没有满足，以及没有满足的原因是什么。假如组织采用的是过程模型，当前的能力级别将是通往下个更高成熟度级别的起点。需要根据组织的需求决定是否继续进行后续的过程改进活动。

测试过程改进的具体形式与组织的发展战略相关。有句话叫“过犹不及”，做事要把握度，不是所有的公司都能像微软那样对质量控制和管理投入大量的财力，也不是所有的项目都需要像 Delphi 第一版发布时那样投入大量的工程师去测试。公司的规模、经济实力、产品投放市场的契机等都将影响测试过程定制的策略，同时也会影响测试过程改进的具体方式和愿意投入的成本，因为商业利润仍然是很多公司做决策的重要依据。

5.4 使用 TMMi 改进测试过程

测试成熟度模型集成 TMMi 框架由 TMMi 协会开发，并作为测试过程改进框架用以对测试过程进行改进。TMMi 也作为 CMMi 1.2 版本的互补模型帮助测试经理、测试工程师和软件质量专家定位某些问题的重要性。像 CMMi 的使用阶段一样，TMMi 也使用成熟度级别概念做测试过程评估和改进，此外还定义了过程域、目标和活动。通过 TMMi 可以使得软件测试从一个无序混乱、缺乏资源、工具和训练有素的测试人员的弱定义过程演变成为以成熟的、可控的，并且以缺陷预防能力为主要目标的，具有完善定义的过程。

TMMi 包含 5 个成熟度等级，旨在补充 CMMi 模型。每一个等级都包括已定义的过程域，组织必须通过实现某个过程域的特殊和通用目标，达到 85%的过程域满足度，才能升级到更高等级。图 5-1 是 TMMi 的 5 个成熟度等级。

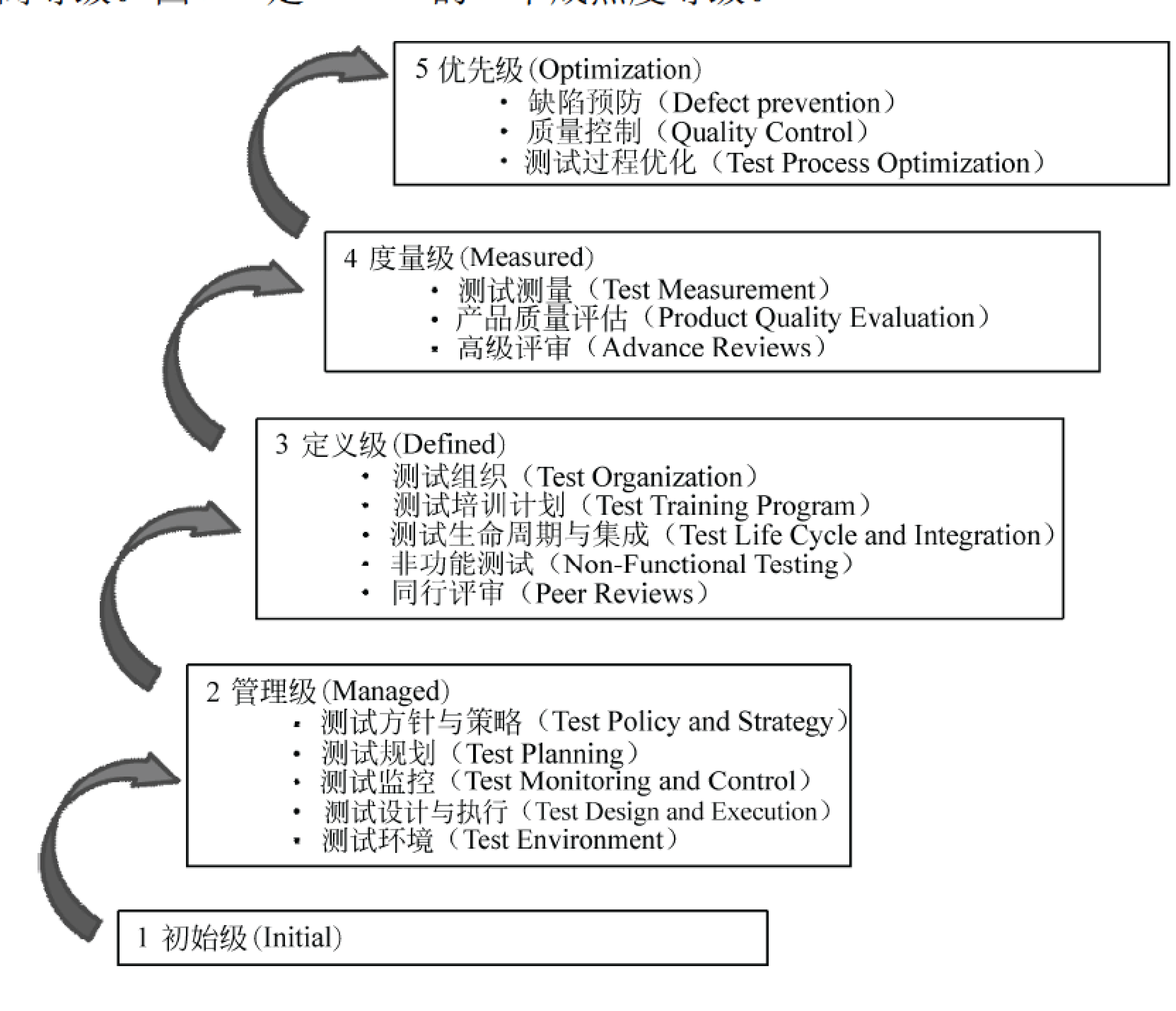


图 5-1 TMMi 的 5 个成熟度等级

请注意，TMMi 中没有针对测试工具或测试自动化专用的特定过程域。因为 TMMi 将测试工具看作是一个辅助资源（实践），作为其所支持的过程域的一部分，例如，测试设计工具作为管理级测试设计和执行过程域的一个辅助测试实践，而性能测试工具作为定义级非功能测试过程域中的一个辅助测试实践。

5.4.1 初始级

TMMi 初始级的特点：测试活动是无序的，甚至是混乱的，没有定义测试过程。测试常常被当作是调试的一部分。组织通常没有提供稳定的环境支持测试过程，测试组织的成功依赖于能力超强英雄式的人物，而不是测试过程。代码完成后测试以一种特殊方式展开，即测试作为调试的一部分去发现和修复软件系统中的缺陷。

初始级的测试目标是软件可以正常地运行，没有严重的失效。在质量和风险没有得到足够评估情况下就发布软件产品。因此软件产品在实际使用过程中，经常会出现不符合需求、不稳定或者响应时间太慢等问题。测试过程中缺少资源、工具和训练有素的测试人员。TMMi 初始级没有定义过程域，且组织的一个特征是倾向于过度承诺，出现严重问题时容易放弃过程，无法重复成功经验。软件产品往往不能按时发布，成本超支且质量不可预料。

5.4.2 管理级

TMMi 管理级将测试从调试中独立出来，并使之成为可管理的过程。管理级的测试过程在实践过程中仍然会有时间压力，很多人仍然觉得测试是编码之后的一个阶段。

管理级的测试过程制定了组织范围或者项目范围的测试策略，并基于此开发测试计划。测试计划中基于产品风险评估的结果定义了测试方法，而风险管理技术用来识别测试依据中的产品风险。测试计划中也定义了哪些测试需要做，什么时候做，谁来做等问题。测试过程中开展测试监控活动，以确保它们能按照测试计划得到执行。假如当前测试状态与测试计划目标之间出现偏差，会有相应的应对措施进行应对。测试过程中应用测试设计技术，从测试依据中设计和选择测试用例。测试工作产品的状态和测试交付物，对管理层而言都是可见的。然而，TMMi 管理级中的测试开始时间还是比较晚，比如通常在设计甚至编码阶段才开始测试活动。

测试过程中包含多个测试级别，例如，单元测试、集成测试、系统测试和验收测试。对于每个测试级别，都制定了组织范围或者项目范围的测试策略，并为每个测试级别制定了特定的测试目标。测试和调试的过程是有区别的。

TMMi 管理级的主要测试目标是检查软件产品是否符合制定的需求，另一个目的是清楚地界定测试和调试的不同角色和职责。管理级的许多质量问题是由于软件测试启动太晚而导致的。缺陷可能从需求阶段、设计阶段到编码阶段引入，由于没有正式的评审去发现和修复这些重要的问题，从而导致很多缺陷遗漏到测试阶段才被发现和修复。另外，管理级的测试活动主要关注在编码之后的测试执行上面。

TMMi 管理级的主要过程域如下：测试方针和策略；测试规划；测试监控；测试设计和执行；测试环境。

5.4.3 定义级

TMMi 定义级不再将测试看成是编码之后的一个阶段，而是集成到整个软件开发生命周期和相关的里程碑。测试计划会在软件项目的初期完成，例如，需求阶段，可能以主测试计划的形式存在。组织定义一套标准的测试过程是定义级的基础，并随着时间不断得到完善。测试过程中明确了测试培训计划，并将测试作为一种职业。尽管测试过程中没有使用正式的评审，但符合定义级的组织需要理解评审在质量控制方面的重要作用，并贯穿于整个软件开发生命周期。TMMi 管理级的测试设计重点关注在功能测试，而定义级需扩展测试技术，将非功能测试包含在这个级别，例如，易用性和可靠性测试。

TMMi 定义级与管理级的一个关键区别是标准、过程描述和规程的范围。管理级在每个项目之间可能采用完全不同的过程，而定义级通常是个别项目或组织单元都只能在裁剪规则的允许范围内对标准过程进行裁剪，因此这些项目有更高的一致性。另一个关键区别是，相对于管理级，定义级在过程描述方面更加严格。因此，定义级的组织必须对管理级的过程域进行重新审视。

TMMi 定义级的主要过程域如下：测试组织；测试培训计划；测试生命周期与集成；非功能测试；同行评审。

5.4.4 度量级

TMMi 的管理级和定义级的目标实现之后，可以具备技术、管理和建立整个测试过程以及测试过程改进支持的基础能力。除了这些，测试还可以成为可度量的过程，从而促进其进一步的发展。度量级的测试是一个完全已定义、具备良好基础的可度量的过程。测试也可以认为是评估过程，它由软件开发生命周期内所有工作产品检查及其相关活动组成。

测试过程中会实施组织范围内的测试度量方案，以评估测试过程质量、效率和有效性，并实施测试过程改进。度量已经纳入组织的测量库，以支持基于事实的决策。测试度量方案还可以用于预测测试能力和成本。

度量级中定义了度量方案，可以使得组织能够通过定义质量需求、质量特性和质量度量实现软件产品质量评价过程。工作产品的评价是使用质量特性的量化指标，例如，可靠性、易用性和可维护性等来实现的。软件产品质量目标在整个软件开发生命周期可用量化术语来理解，并针对已定义的目标来进行管理。

评审作为测试过程的重要组成部分，用来在软件开发生命周期早期测量产品质量，并作为正式控制质量的阶段点。同行评审作为缺陷检测技术，变成与产品质量评估过程域保持一致的产品质量测量技术。

度量级还包含：建立同行评审（静态测试）和动态测试之间协作的测试途径，使用同行评审结论和数据来优化测试途径，这些都是为了使测试更有效率和有效果。同行评审已完全与动态测试过程集成，例如一部分的测试策略，测试计划和测试途径。

TMMi 度量级的主要过程域如下：测试度量；产品质量评估；高级评审。

5.4.5　优化级

TMMi 从初始级到度量级，所有测试过程改进目标的实现都为测试创造了一个组织的基础架构，即支持完全的已定义和已测量的过程。TMMi 优化级基于统计数据控制过程的定量认知，具备了持续过程改进的能力。提高测试过程能力是通过过程和技术增量和创新改进进行的，同时优化测试方法和技术，并持续关注细微调整和过程改进。

什么是优化的测试过程，TMMi 中定义为：已管理的、已定义的、已测量的、有效率和有效果的；统计控制的和可预测的；关注于缺陷预防；自动化支持被视为资源的有效利用；能够支持技术从行业转移到组织；能够支持测试资产的重复使用；专注于过程改变，实现持续改进。

为了支持测试过程基础架构的持续改进，并识别、计划和实现测试改进，通常会正式成立永久的测试过程改进小组，小组成员都接受过能提高他们技能的专业训练，从而获得帮助组织成功所需的技能和知识，很多组织中称之为测试过程组（TPG）。在 TMMi 定义级，引入测试组织时即可开始正式支持测试过程组。随着在度量级和优化级中引入更多高级别的实践，其职责也增加了，例如，确定可重用的测试过程资产，开发和维护测试过程资产库。

建立缺陷预防过程域，是为了识别和分析在软件开发生命周期中出现的缺陷的根本原因，并制定措施以防止今后再发生类似的缺陷。测试过程性能的异常，是过程质量控制的一部分，对它们进行分析，以查明它们的原因，可以作为缺陷预防的一部分。

目前，测试过程通过质量控制过程域进行统计管理，包括统计抽样、测量置信水平、可信度和可靠性驱动测试过程。测试过程的特点是基于抽样的质量测量。

优化级中，测试过程优化过程域引入了微调机制，不断改进测试。有一个既定的规程来识别过程改进，同时也通过选择和评价新的测试技术来识别过程改进。支持测试过程的工具，在以下方面都起到了作用：测试设计、测试执行、回归测试、测试用例管理、缺陷收集和分析等。组织中的过程和测试件的重复使用也是常见的做法，并由测试（过程）资产库支持。

优化级的三个过程域：缺陷预防、质量控制和测试过程优化都为持续过程改进提供支持。事实上，这三个过程域是高度相关联的。例如，缺陷预防过程域支持质量控制过程域，如分析过程性能的异常值和进行缺陷因果分析，并实施预防缺陷再次发生的实践。质量控制过程域有助于测试过程优化过程域，测试过程优化过程域支持缺陷预防过程域和质量控制过程域，例如，通过实施测试改进建议来支持缺陷预防过程域和质量控制过程域。所有

这些过程域，依次需要低级别过程域完成时所获得的实践来支持。在优化级，测试是一个以预防缺陷为目的的过程。

TMMi 优化级的主要过程域如下：缺陷预防；质量控制；测试过程优化。

更多 TMMi 的内容，可以参见 www.tmmi.org。

5.5 使用 TPI-Next 改进测试过程

TPI-Next 是领先的改进测试过程方法。TPI-Next 通过分析当前测试过程的状况，获取其中的优点和弱点。该模型也可以用来评估和构建测试过程改进的特定目标，同时为达到这些目标提供改进的路线图。

TPI-Next 模型的结构如图 5-2 所示。

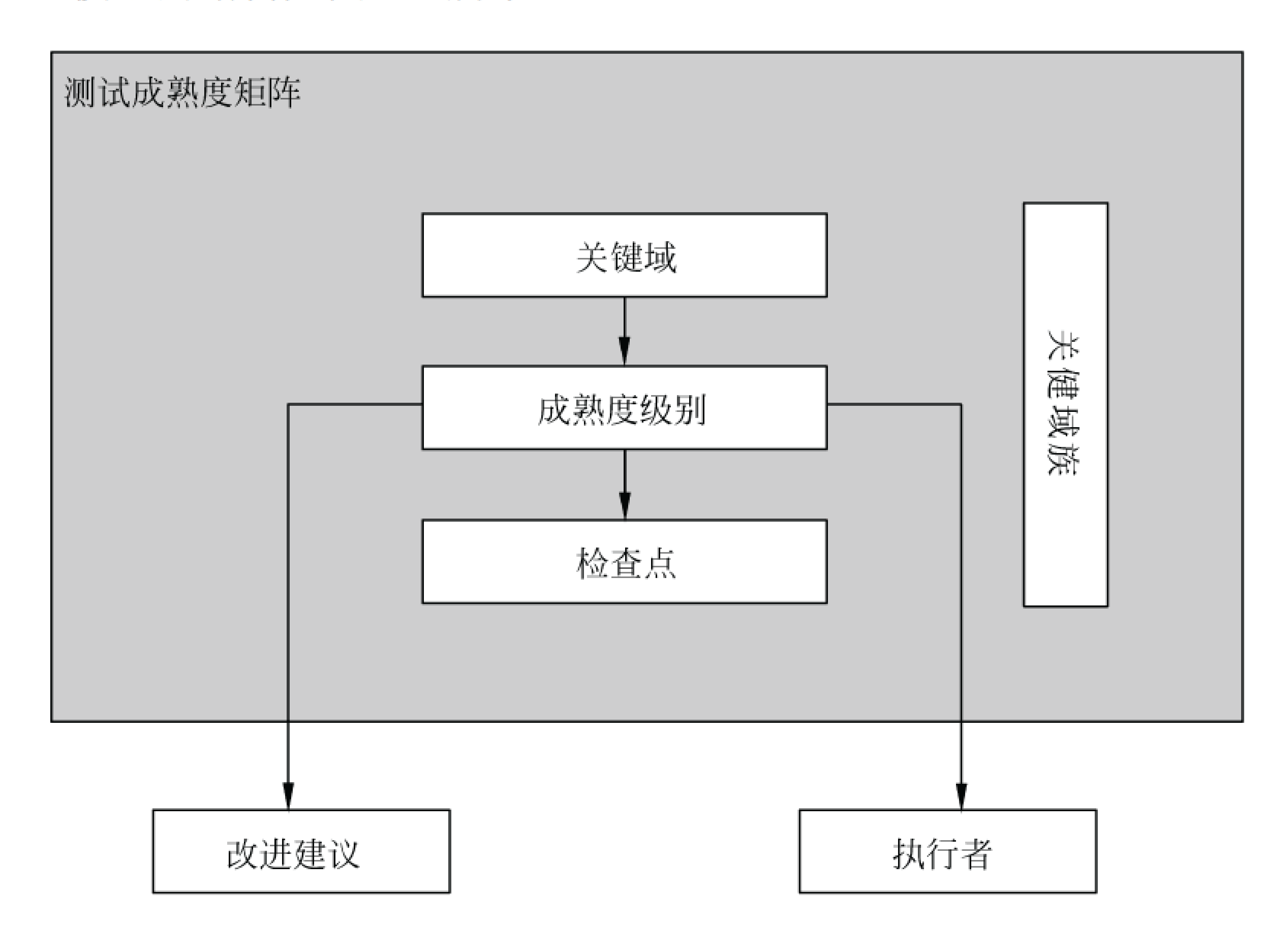


图 5-2 TPI-Next 模型

测试过程是由支持不同测试活动的关键域组成，TPI-Next 模型由 16 个关键域组成。每个关键域有不同的成熟度级别，而不同关键域的组合定义了作为整体的测试过程的成熟度级别。TPI-Next 模型的 16 个关键域分别如下。

（1）利益干系人承诺（Stakeholder Commitment）；

（2）参与程度（Degree of Involvement）；

（3）测试策略（Test Strategy）；

（4）测试组织（Test Organization）；

（5）沟通（Communication）；

（6）汇报（Reporting）；

（7）测试过程管理（Test Process Management）；

（8）估算和计划（Estimating and Planning）；

（9）度量（Metrics）；

（10）缺陷管理（Defect Management）；

（11）测试件管理（Testware Management）；

（12）方法论实践（Methodology Practice）；

（13）测试人员专业性（Tester Professionalism）；

（14）测试用例设计（Test Case Design）；

（15）测试工具（Test Tools）；

（16）测试环境（Test Environment）。

TPI-Next 模型将成熟度级别分为 4 层，分别是：初始级、控制级、高效级和优化级。只有达到前面的成熟度级别，才能完成当前的成熟度级别，并进入下一个成熟度级别。而初始级比较特殊，该成熟度级别没有包含任何的特定关键域和预期期望结果，相当于任何测试过程都是自然存在该级别的。

TPI-Next 模型中的每个成熟度级别，针对测试过程的预期期望都是通过检查点来进行定义的，检查点就是该模型的测度单元。TPI-Next 检查点提供了针对测试活动的“是”与“否”的判断句子。针对特定的测试过程，假如基于充分的信息，其测试点的回答是“是”，说明针对该测试过程的关键域，满足了该成熟度级别的要求。每个检查点都与特定的关键域与成熟度级别相关联。假如与关键域关联的检查点的回答都是“是”，则说明该关键域达到了特定的成熟度级别。

假如与某个测试过程相关的所有关键域都已经达到特定成熟度级别，那么作为整体的该测试过程也达到了该成熟度级别。例如，如果所有相关的关键域都达到了控制级的最低成熟度级别的要求，那么就可以认为整个测试过程达到了控制级。类似地，如果所有相关的关键域达到指定的成熟度级别，整个测试过程即可以达到高效级或者优化级。

TPI-Next 模型提供了测试过程渐进式的改进方式，从初始级、控制级、高效级一直到最终的优化级。而每一个步骤都通过检查点组提供了改进的建议。检查点组从多个关键域中集成了一组检查点，作为过程改进步骤中的一个检查列表，其目的是为了加强测试过程成熟度。每个检查点组以字母表示其在测试过程改进路径中的位置，例如，A 表示过程改进的第一步。

TPI-Next 模型还提供了另外两个元素：改进建议和执行者。他们为加快测试过程成熟度提供了额外的信息。改进建议关注在测试过程本身。基于多年使用模型经验，TPI-Next 模型提供了许多最佳实践。而执行者显示了软件开发生命周期中，测试过程与其他过程可以从彼此的最佳实践中获益。它们回答了如下问题：测试如何从其他过程活动中获益？或者其他过程活动如何从测试中获益？例如：

（1）通过使用配置管理改进测试件管理；

（2）缺陷管理提供了基于根本原因分析的问题管理，并为组织发现弱项提供支持；

（3）通过使用需求管理改进测试依据质量。

已经采用软件过程改进（例如，CMMi 或者 SPICE）的组织，同样可以从上面提到的参与者中得到支持。更多关于 TPI-Next 的内容，可以参考 www.tpinext.com。

5.6 使用 CTP 改进测试过程

使用核心测试过程（Critical Test Process，CTP）评估模型的基本前提是了解特定的测试过程。假如核心过程能够实施良好，将会造就成功的测试团队。相反，如果这些活动实施的程度不够，即使是有天赋的测试人员和测试经理也很难获得成功。

5.6.1 模型结构

什么是核心测试过程？简言之，假如一个过程满足下面的一个或多个标准的时候，它就变成了核心测试过程。

（1）作为必不可少的测试工作的一部分，测试团队经常性地重复这个过程。有效地处理重复发生的过程有助于保证日常测试活动的高效性和一致性。例如，缺陷报告过程作为测试实施的后续过程反复发生。

（2）影响测试人员协同工作能力的过程，特别是该过程失败的结果可能不利于测试团队的团结和合作。例如，当测试团队正在运行一组测试用例时，这个过程应该支持一个具体的测试任务分派方式，用以防止偶然因素导致测试人员进行重复工作。

（3）由同级或高级管理人员参与的过程。测试团队活动和输出的高可见性，有利于测试团队在项目中的声誉，也有利于得到项目中其他团队的认可。例如，当报告测试状态时，测试人员必须向同级或者高级管理人员展示测试的结果，并能够用有效的方式交流已经完成的测试、发现的问题和测试的价值等方面的信息。

（4）过程的失败可能导致严重和长期的负面后果。例如，在识别测试条件的时候，由于选择了不合适的测试目标，漏掉对客户至关重要的功能的测试，会增加产品的质量风险，对于特定类型的系统，甚至还可能对客户产生意料不到的人身伤害。

测试的核心过程直接影响了测试团队在它的组织、操作和技术环境中提供有价值的信息和服务的能力。如果测试人员和测试经理掌握并不断改进这样的过程，那么就能够提供更好的信息和服务。该模型共识别了包括测试过程在内的 12 个关键测试过程，如图 5-3 所示。

核心测试过程（CTP）主要包括以下过程。

（1）测试过程：测试过程是对其他 11 个过程的综述，如图 5-3 所示，包括计划、准备、执行和完善 4 个部分的活动。具体内容包括：理解测试工作、配置人员和测试、运行测试并收集结果，以及系统不断适应和改进。

（2）发现测试场景：了解组织中软件开发、维护或者采购的整个过程；研究测试范围、

测试工作产品、开发工作产品、质量相关文档、数据和度量方法等；与其他的利益干系人讨论他们正在进行的测试活动，以及在测试团队参加进来之后，他们会继续进行哪些活动；确定测试过程中同级别的利益干系人；与经理以及其他高级经理讨论，确定测试团队要创造的价值，特别是测试在质量保证中充当的角色。

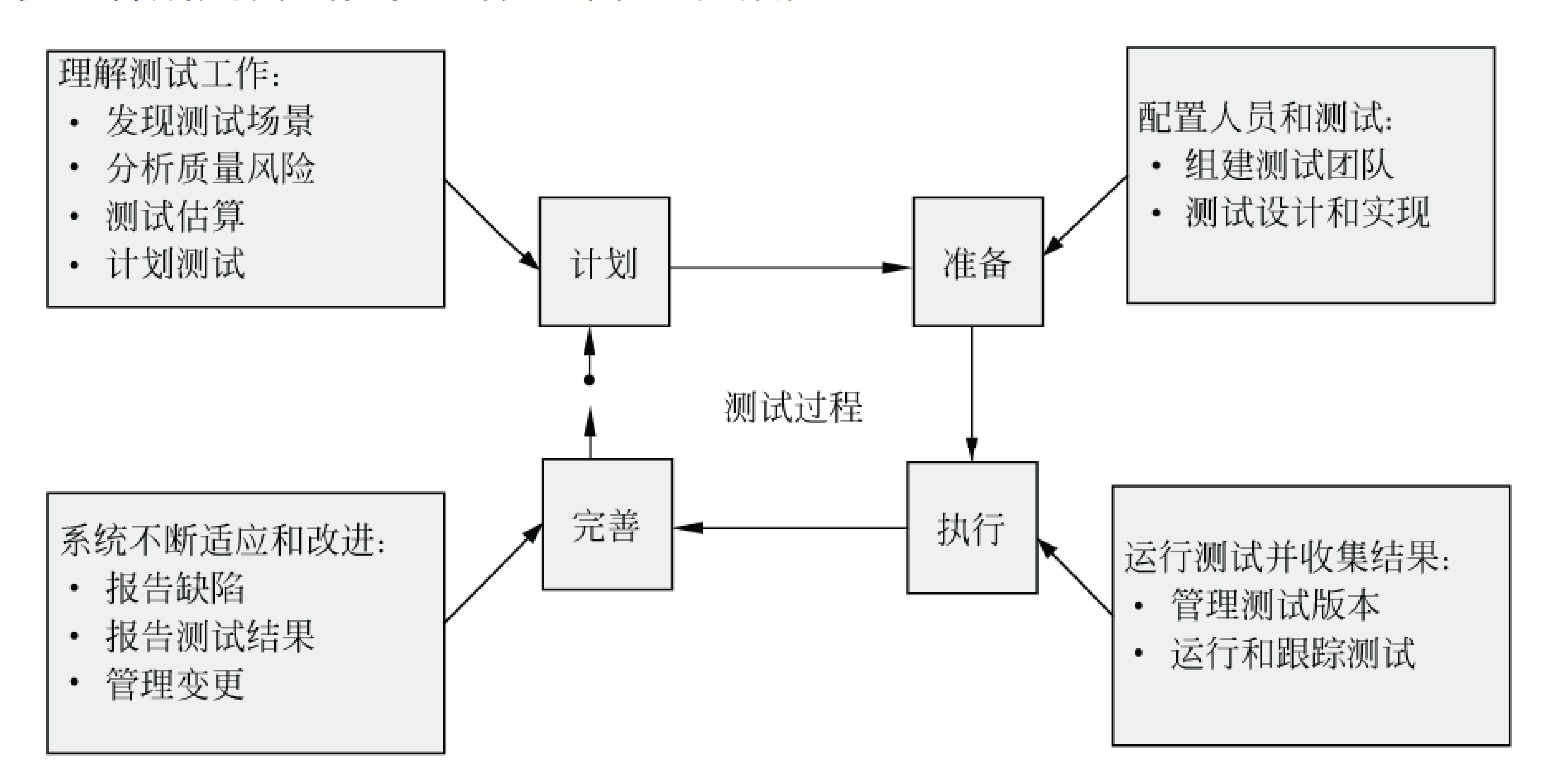


图 5-3 CTP 测试过程

（3）分析质量风险：找出关键的测试和质量利益干系人，获得利益干系人参与质量风险分析的承诺；就质量风险分析的技术和方法问题，对关键利益干系人进行调查；向关键利益干系人收集关于质量风险、相关的失效模式以及这些失效对于质量的影响和风险的优先级方面的意见；以文档形式记录对应于所使用技术的质量风险；将质量风险分析文档录入项目的资料库或配置管理系统。

（4）测试估算：开发工作分解结构并估算进度；使用工作分解结构和进度表生成预算；获得管理层对估算的进度表和经费预算的支持；将得到批准的经费预算和进度表的文档录入项目的资料库或配置管理系统。

（5）计划测试：研究、设计、收集测试子项目的战略、战术和内部工作方式，并编写相关的文档；商议在测试子项目和总体项目之间的协调工作，并编写相关的文档；确定剩余的后勤工作和计划的细节，并编写文档；发布该计划并提供给利益干系人进行评审；将测试计划录入到项目的资料库或配置管理系统。

（6）组建测试团队：招聘合适的测试团队成员；提高测试团队的技能，明确团队成员的职业发展方向。

（7）测试设计和实现：选择合适的测试套件，并在最重要的还没有覆盖的风险区域识别相关测试条件；根据将要覆盖的测试条件、期望结果、可用的时间和预算、被测试系统的可测试性、测试环境和测试人员具有的技能等选择合适的测试技术；使用静态技术和动态技术对系统进行测试；检查存放在产品库或配置管理系统中的测试用例、测试规程规格说明、测试环境的配置信息和任何其他方式生成的文档，将测试件的版本与待测试系统的版本链接起来；根据测试设计和实现过程中学到的知识更新质量风险列表，评估测试对于

质量风险的覆盖率。

（8）管理测试版本：管理因变更（例如：修复缺陷和引入新功能）造成的版本修改，当完成了修改并通过组件测试之后，将这些修改存储到配置管理系统；在配置管理系统中取出源文件进行编译、链接、汇编这些构件，并用特定的版本号标记这些构件；将构件制作成可以安装的文件，并将它提交给在测试环境中负责安装这个软件的人进行安装；在测试环境中对该构件进行冒烟测试。

（9）运行和跟踪测试：根据风险的优先级、项目的限制和任何其他恰当的考虑，选择应该在本次测试周期中运行的测试套件（从测试集中选择）；把测试套件中的测试用例分派给测试人员执行测试；不断地执行这些测试用例，报告错误并捕获关于测试的信息；当发现阻塞问题时，解决这些问题；每天报告测试的状态，调整任务的分派，并考虑计划和优先级；如果合适的话，按照优先级的次序，删除不可实现的或冗余的测试（先删除优先级最低的测试）；周期性地报告测试的发现和状态；检查所有状态的文档、初始文档，更新测试用例或其他测试系统的元素，或其他存储在项目的资料库或配置管理系统中的记录。

（10）报告缺陷：使用结构化的测试技术运行每个测试；重现任何与期望的行为不符的偏差；通过检查主要变化的影响来隔离错误，并寻找绕开问题的方法；寻找通用案例，特别是更严重的情况；与相似的条件下以前版本的系统行为做比较，观察异常行为；总结失败，包括对于客户或用户的影响；精简缺陷报告，删除不必要的信息；表达清楚，删除混淆的、误导的或不精确的用词；综合处理缺陷报告，合理和公平地表达意见；提交缺陷报告供同行复查，解决发现的所有问题，并将缺陷报告录入到缺陷跟踪系统。

（11）报告测试结果：了解测试报告的听众，通常包括所有测试过程和系统质量的利益干系人；定义要展示的结果，一般是能够回答利益干系人将对测试提出的问题的信息，特别是测试结果对于项目目标的意义；选择度量，并构建能够回答这些问题的报告和图表；根据需要向利益干系人展示测试的结果。

（12）管理变更：收集变更请求和缺陷报告；在变更控制委员会会议上对提出的请求进行评审；计划、实现、测试和集成相关的变更请求，注意新的成本、收益或风险；展示实现、测试、分布集成结果和可交付物，以得到最后的批准；如果批准了包含的修改，将新的或改变了的系统组件、项目文档和其他交付物录入配置管理系统。

CTP 中每个核心测试过程都包括详细的实施步骤。表 5-3 是第一个核心过程（测试过程）的详细步骤。

表 5-3 测试过程的实施步骤

步骤序号	步 骤 说 明	完成否
1	计划：理解测试工作	
1.A	理解测试运作的（系统、项目和过程）背景和组织背景	
1.B	定义系统质量的风险，列出优先级，并且让利益干系人认同测试起到的降低风险的作用	
1.C	对执行步骤 1.B 中的测试所需要的时间、资源和预算进行评估，并获得管理层的支持	

续表

步骤序号	步 骤 说 明	完成否
1.D	开发一个计划，以确定降低系统质量风险所需要的活动、依赖性和参与者，并且取得利益干系人对该计划的支持	
2	准备：配置人员和测试	
2.A	通过人员配备和培训，组建一个专业的测试团队	
2.B	设计、开发、采购和验证测试系统，测试团队利用这个测试系统评估待测系统的质量	
3	执行：进行测试并收集结果	
3.A	获取并且安装测试版本，包括待测系统的某些或所有组件	
3.B	分配、跟踪并管理用于每个测试版本的测试用例	
4	完善：指导系统不断适应和改进	
4.A	记录测试过程中发现的缺陷	
4.B	与关键利益干系人交流测试结果	
4.C	根据变化进行调整，优化测试过程	

5.6.2　评估模型

使用 CTP 模型对测试过程进行改进是比较灵活的。CTP 的 12 个核心测试过程构成了测试过程改进的基础，但是该模型并不要求同时对这 12 个测试过程进行改进，可以根据实施过程改进的测试团队的实际情况挑选 12 个核心测试过程中的任意一个或多个过程进行改进。使用 CTP 的过程改进，始于对现有测试过程的评估。评估识别了过程的强弱项，并结合组织的需要提供改进的意见。

CTP 并没有严格定义每个过程的评估标准和度量指标，具体的评估标准可以根据组织和每个测试团队的实际情况进行定制。但是 CTP 对评估给出了一些建议。评估时最常用的 5 个问题如下。

（1）测试是否有效（测试是否发现了重要的缺陷，而不是客户发现了它们）？

（2）测试的效率如何（测试的投资回报）？

（3）测试介入软件开发的时间（测试和质量风险管理是否从项目的第一天就开始）？

（4）测试是否在为项目经理的决策及时提供了相关信息？

（5）测试过程和测试团队成员是否能够持续地改进和提高？[①]

针对上面列出的这些常见的评估问题，可以采用定量和定性两种度量数据进行分析。一般而言，CTP 评估将对下列相关的度量数据进行分析。

（1）缺陷发现率；

（2）测试投资回报率；

① Rex Black. Critical Testing Processes. www.rexblackconsulting.com，2003.

（3）需求覆盖率和风险覆盖率；
（4）测试发布管理费用；
（5）缺陷报告拒绝率。
除了上述的度量数据外，CTP 评估过程通常还考虑下面的质量因素。
（1）测试团队角色和效率；
（2）测试计划有效性；
（3）测试团队测试水平、背景知识和技术；
（4）缺陷报告效率；
（5）测试结果报告效率；
（6）变更管理效率。
CTP 评估过程中识别出薄弱过程域后，需要制订改进计划，对薄弱的过程进行改进。

5.7 使用 STEP 改进测试过程

系统化测试和评估过程（Systematic Test and Evaluation Process，STEP）定义了软件评估的主要活动。[①] 评估作为软件工程的一个分支，关注的是如何判断软件产品做了它应该做的事情。软件评估采用的主要技术包括分析、评审和测试。其中，测试作为最复杂的一种技术，成为 STEP 关注的重点。不过，协调和计划这三种技术是实施过程改进的前提。STEP 首先强调测试中的缺陷预防功能，然后将缺陷检测和软件能力验证作为第二个目标。

5.7.1 组成

STEP 模型由特定的阶段（活动或任务）、工作产品（文档和测试用例）和角色（为任务定义相应的角色和职责）组成，可以有效地帮助软件产品达到质量的要求。图 5-4 是 STEP 的组成元素和关系图。

5.7.2 架构

STEP 架构描述了测试计划阶段如何将测试活动分解为不同的级别，如图 5-5 所示。每个级别都代表了特定的测试环境（例如，组件测试通常是开发人员在开发环境下进行的软件测试）。对于简单的项目，或者系统简单的功能升级，可能只需要一或两个测试级别就可以，例如，组件测试和系统测试。而对于复杂的项目，就需要更多的测试级别，例如，组件测试、集成测试、系统测试和验收测试。

① Rick D Craig，Stefan P Jaskiel. Systematic Software Testing. Artech House，2002.

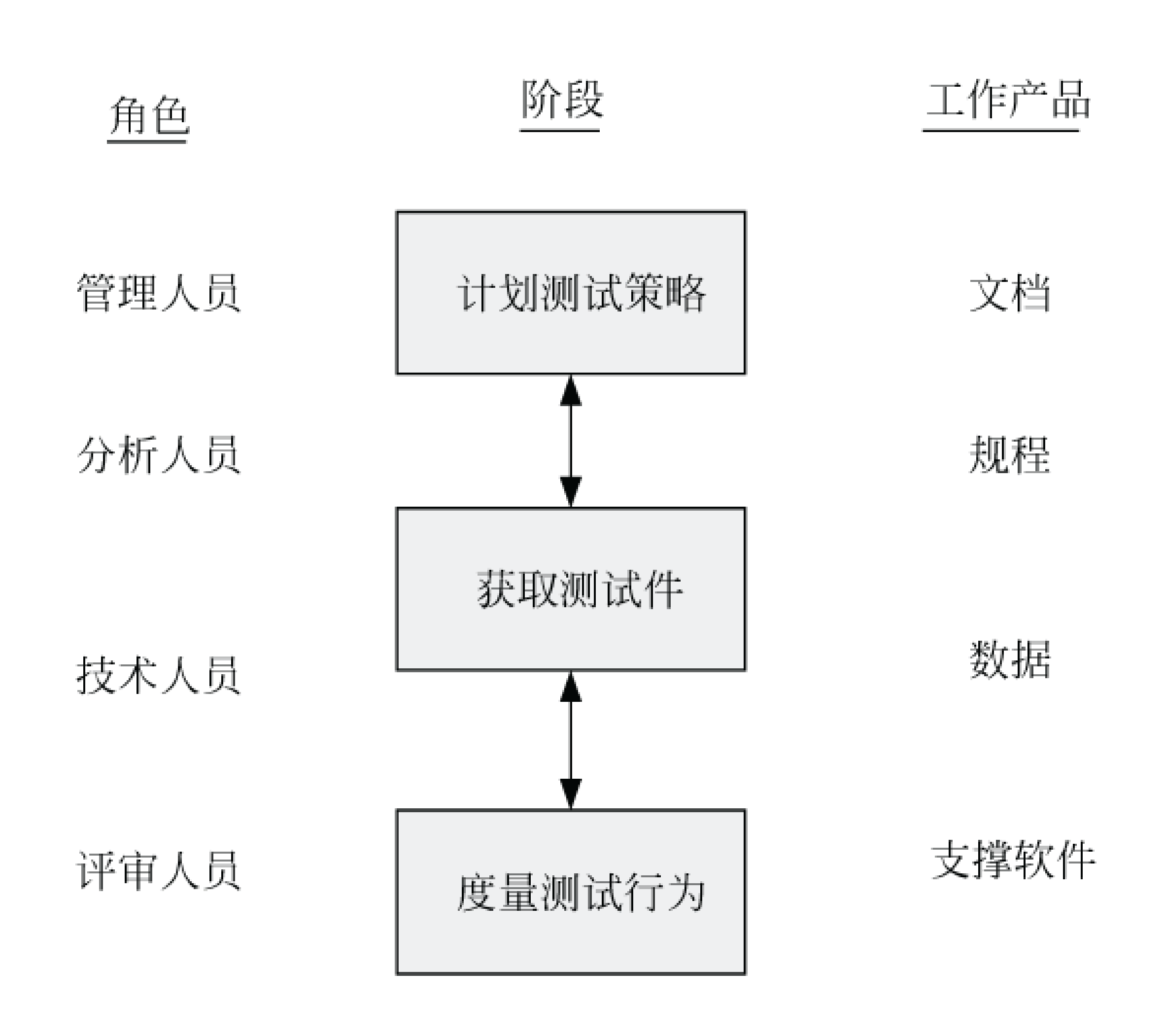


图 5-4　STEP 的组成及关系

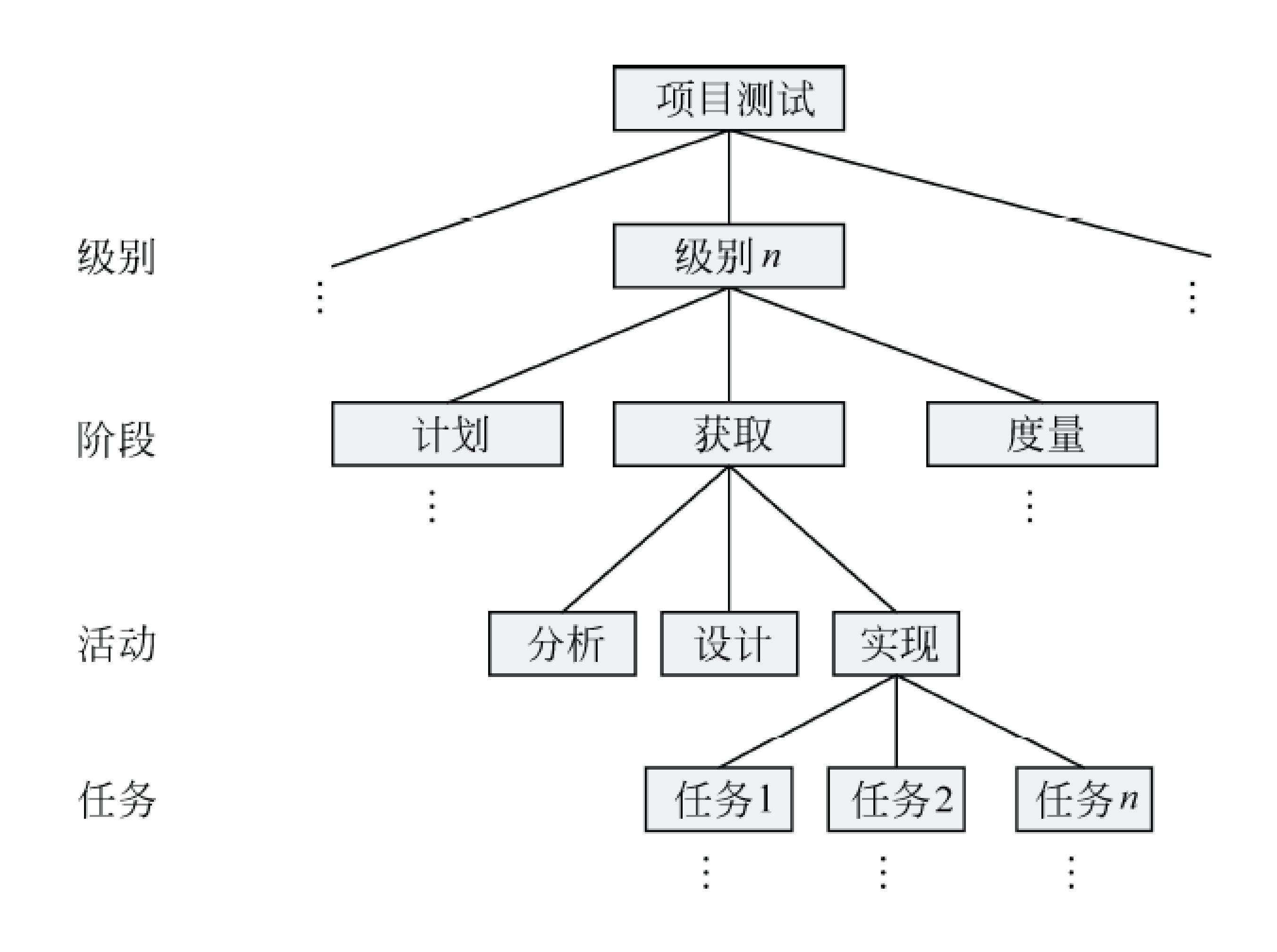


图 5-5　STEP 的架构

STEP 提供了一个模型，可以作为建立详细测试计划的开始点。模型中所有的组成元素都可以进行合理的裁减和修改，或者扩展以满足实际的测试需求。STEP 由三个阶段组成，分别是计划测试策略（选择测试策略和规定测试级别和方法）、获取测试件（描述详细的测试目标、设计和实现测试用例）和度量测试行为（执行测试用例和评估软件和过程）。这三个阶段可以应用到任何一个级别。STEP 的三个阶段可以进一步分解为 8 个主要的测试活动，如表 5-4 所示。

5.7.3　活动时序

STEP 规定了实施测试活动和任务的时间点，以及具体的先后顺序。STEP 测试活动的时序强调的是大部分的测试设计工作应该在软件详细设计之前完成。图 5-6 是各个测试级

别的 STEP 测试活动时序。

表 5-4 STEP 的阶段和活动

阶段 1	**计划测试策略**
P1	建立主测试计划
P2	开发级别测试计划
阶段 2	**获取测试件**
A1	收集测试目标
A2	设计测试用例
A3	实现测试计划和设计
阶段 3	**度量测试行为**
M1	执行测试用例
M2	检查测试是否充分
M3	评估软件对象和测试过程

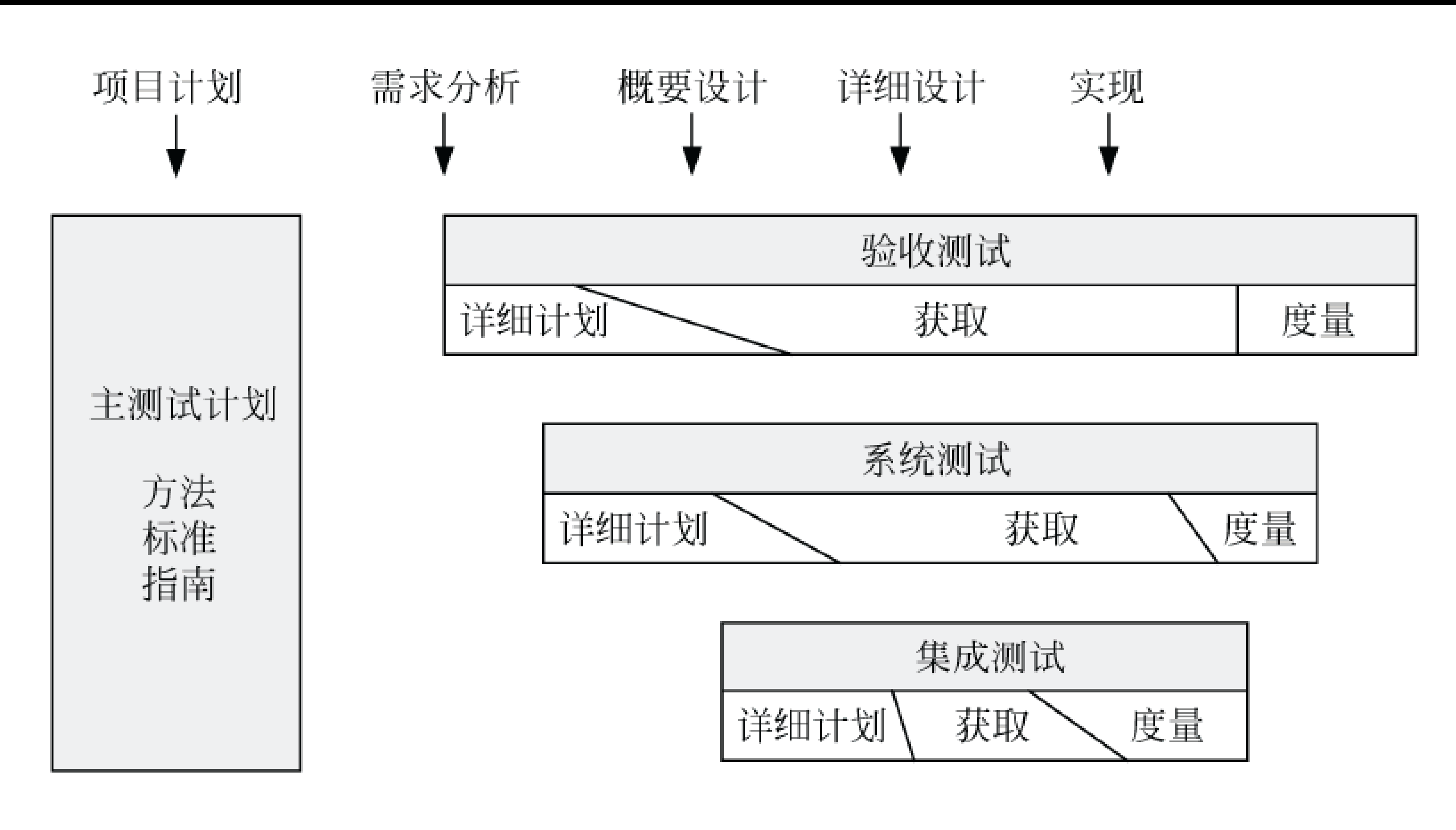


图 5-6 STEP 的测试活动时序

测试设计会随着详细的软件设计而持续进行，同时根据软件的详细设计识别新的测试条件，增加新的测试用例。随着软件详细设计的继续深入，例如，针对软件组件和模块的设计文档成型之后，相应的测试设计也会继续深入到组件和模块级别。假如软件项目处于编码阶段，相应的测试设计也会基于代码和具体的实现开展。

测试识别和设计活动会在不同的测试级别重叠。每个测试级别的目标是尽快完成本级别的测试设计，这个目标可以帮助检查相应的需求是不是满足可测性要求，同时可以在过程的早期发现和修复缺陷。因此，这样的策略可以开展有效的软件评审。

度量测试行为阶段也是基于不同的测试级别开展的。组件测试首先执行，然后将不同的模块和组件进行集成，最后开始相应的系统测试和验收测试。执行顺序从低测试级别到高测试级别是必须遵循的一个限制，而对于测试计划和获取测试件的阶段，并没有这么严格的要求。可以首先设计高级别的测试件，尽管它们在后期才会使用到。

图 5-7 是针对某个测试级别的测试活动时序图。通常来说，首先确定测试计划和测试目标，然后开始测试设计和实现，最后才是测试执行和评估。不同的测试活动之间是可能存在重叠的。

5.7.4　工作产品

STEP 过程模型的另外一个重点是每个阶段和活动中生成的工作产品。STEP 将测试工作产品通称为测试件，例如，测试计划和测试设计规格说明等。测试件是和软件类似的产品，如图 5-8 所示。从中可以看出，测试活动和开发活动是并行进行的。软件经历了计划、设计和实现，同样，测试件也经历了计划、设计和实现。

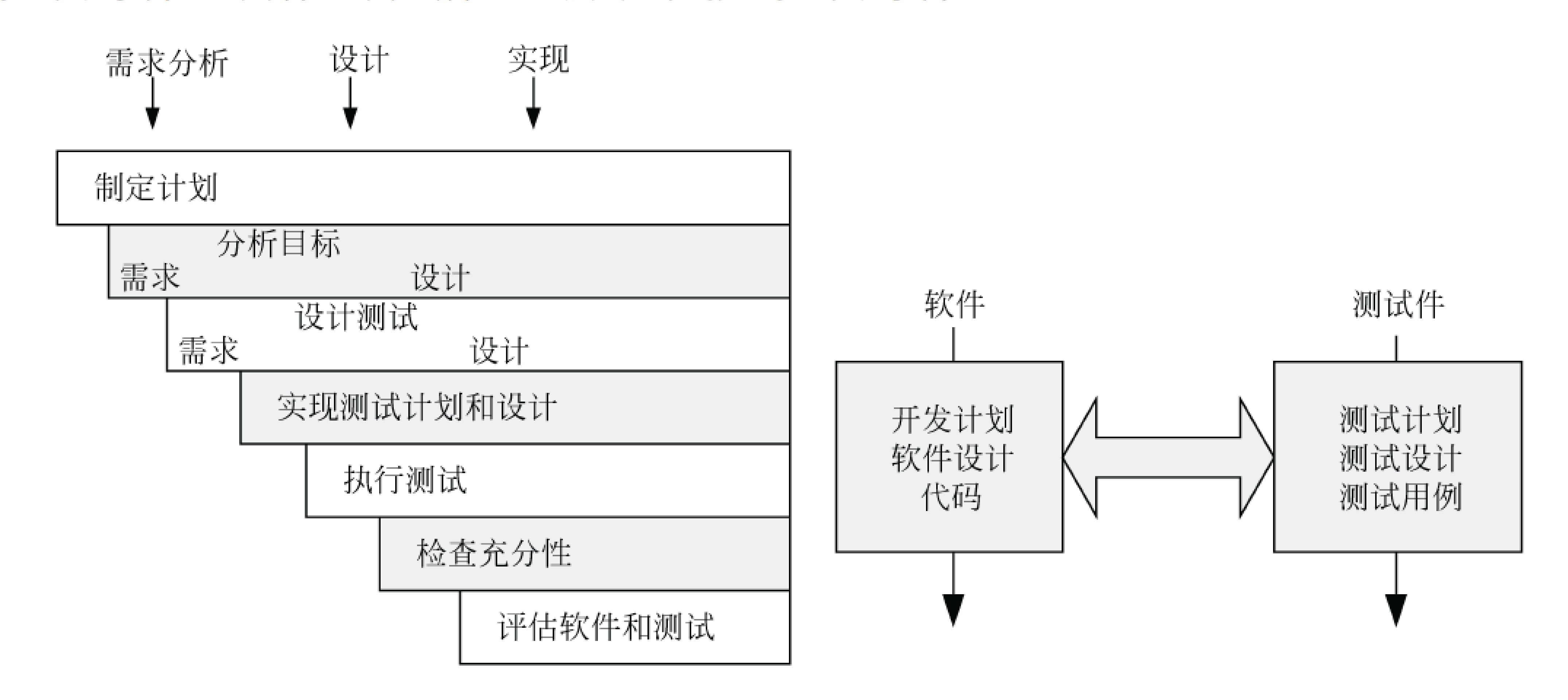


图 5-7　某测试级别的测试活动时序　　　　图 5-8　软件和测试件

软件开发和测试之间的工作产品是相互支持的。测试件的开发依赖于软件开发的工作产品，可以帮助预防和检测软件中存在的缺陷；软件的开发通过评审测试工作产品，可以帮助预防和检测测试件中的缺陷。

STEP 的工作产品是基于 IEEE Std 829—2008 的，标准中定义的主要测试工作产品如表 5-5 所示。

表 5-5　软件测试文档

序　号	测 试 文 档
1	测试计划：用于主测试计划和级别测试计划
2	测试设计规格说明：用于各个测试级别，描述测试集的架构和覆盖可追溯性
3	测试用例规格说明：用来描述测试用例或者自动化测试脚本
4	测试规程规格说明：用来描述测试用例集的执行顺序
5	测试日志：用来记录测试规程执行的信息
6	测试事件（缺陷）报告：用来描述测试或者产品中发生的异常。这些异常可能来自需求、设计、代码、文档或者测试用例
7	测试总结报告：用来报告某个测试级别或整个测试的完成情况

5.7.5　角色和职责

STEP 定义了不同测试活动相关的角色和职责。STEP 中的 4 个主要角色分别是管理人员、分析人员、技术人员和评审人员，他们的职责描述如下。

（1）管理人员：沟通、计划和协调。
（2）分析人员：计划、识别、设计和评估。
（3）技术人员：实现、执行和检查。
（4）评审人员：检查和评估。

管理人员主要负责测试的计划和协调，和项目中的其他利益干系人进行相关信息的沟通；测试分析人员主要负责识别测试目标和需要覆盖的范围、测试设计、测试评审和评估。技术人员的主要职责是根据测试分析人员提供的设计，实现测试规程和测试集、执行测试、检查测试结果并和测试出口准则进行比较、编写测试日志并报告缺陷；测试评审人员对测试过程和测试过程中的工作产品进行评审。

STEP 并不要求这 4 个角色由不同的人员担当。在规模较小的项目中，一个人可能承担了这 4 个角色和职责。而对于大的项目和成熟度较高的组织，4 个角色可能由不同的人来担当，并且为不同的角色定义不同的职业规划。

小结

过程改进与软件开发过程和测试过程有着密切的联系。吸取以前的经验教训有助于组织的过程改进，而软件开发和测试也能在这个过程当中受益。戴明改进循环 PDCA（Deming Improvement Cycle）：计划（Plan）、实施（Do）、检查（Check）和行动（Act），已经在业界使用了多年。除此之外测试团队也可以采用本节描述的 IDEAL 来实施测试过程的改进。

为了达到更高的成熟度和更专业的水平，IT 行业已经开始参照测试过程改进模型开展工作。行业标准模型有助于开发跨组织的度量与分析。过程参照模型（例如 TMMi 和 CMMi）提供的比较标准供不同的公司和组织使用；内容参照模型允许组织拥有更多的自由度，根据所确定的优先级实施过程改进。针对测试领域的过程改进需要，出现了多个过程改进的标准和模型，包括 STEP、TMMi、TPI-Next 和 CTP 等。

对于不同的测试过程评价模型，组织应根据目前的测试过程和团队能力水平，确定自身的能力成熟度。TMMi 和 TPI-Next 为测试过程的改进提供了明确的操作步骤。STEP 和 CTP 则是向组织提供方法以判定哪些过程改进能带来最大的回报，并让组织选择合适的路径。CMMi 作为一个优秀的成熟度模型，有良好的架构和评估过程，但是除了测试之外，它还包括很多其他的内容。

模拟题

40．CTAL-ATM_LO-5.2.1

TM-5.2.1(K2) 举例说明测试过程改进的重要性。

问题：

关于改进测试过程的重要性，下面哪两个论述相对而言是最好的例子？

选项：

A．由于测试常常占用了总体项目成本的很大份额，更有效的测试可以促进项目更有效
B．测试过程改进模型可以帮助达到更高级别的成熟度和专业化
C．由于测试常常占用了总体项目成本的很大份额，因此软件过程改进模型更注重测试过程，例如：CMMI
D．假如使用测试过程改进模型，测试人员需要改进测试过程时，戴明（Deming）改进环：计划、实施、检查和行动，与它是没有关系的
E．测试过程改进很重要，存在众所周知和行业已经接受的测试过程改进模型，例如 TMMi、TPI-Next 或者 CTP

解释：

A．正确。参考大纲 5.2 节。
B．正确。参考大纲 5.3 节。
C．不正确。与大纲建议相矛盾。
D．不正确。与大纲建议相矛盾。
E．不正确。错误的结论。
分值：1 分

41．CTAL-ATM_LO-5.3.1

TM-5.3.1(K3)使用 IDEAL 模型定义测试过程改进计划。

问题：

假设你是测试经理，正在设法让测试过程更加有效和高效。针对该过程改进，管理层已经批准了初始的预算。上个星期，外部的咨询师完成了她的评估，并提交了她的发现。

假如你的过程改进遵循 IDEAL 模型，下面哪三项是过程改进的剩余（最后三个）步骤？

选项：

A．创建计划来选择和实施评估的建议
B．实施评估的建议，包括必要的培训和试点项目
C．从改进中评估收益，包括投资回报率
D．在整个测试组织内启动改进过程
E．用一系列内部优先事项来取代咨询师的建议
F．通过评估低效的根源以诊断当前的状况
G．采取步骤将组织的测试过程提升到成熟度级别 5

解释：

A．正确。IDEAL 过程的第 3 个步骤。
B．正确。IDEAL 过程的第 4 个步骤。
C．正确。IDEAL 过程的第 5 个步骤。
D．不正确。这是已经实施的 IDEAL 过程的第 1 个步骤。
E．不正确。尽管这会经常发生。
F．不正确。这是已经实施的 IDEAL 过程的第 2 个步骤。
G．不正确。题目中并没有假设组织遵循 TMMi。
分值：3 分

42．CTAL-ATM_LO-5.4.1

TM-5.4.1(K2) 总结 TMMi 测试过程改进模型的背景、范围和目标。

问题：

你在国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

你的产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少已经工作了两年时间，大部分人都具有更长的工作时间。经验丰富的测试人员利用公司内部的定制测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。你的公司已在公司级别实施了 CMMi 三级。

团队在赶上硬件开发进度方面存在问题。你认为 TMMi 可以帮助改善项目。下面 TMMi 的哪个方面最适合满足该目标？

选项：

A．将测试改进与公司层面的改进相匹配
B．达到优化级以帮助预防缺陷
C．从初始级提升到管理级
D．完成 85%特定和通用的目标

解释：

A．正确。TMMi 支持 CMMi，这是公司的选择。
B．不正确。基于场景信息，你不太可能是这么高的级别。
C．不正确。基于场景信息，你不太可能是这么低的级别。
D．不正确。不相干的详细信息。
分值：1 分

43．CTAL-ATM _LO-5.5.1

TM-5.5.1(K2) 总结 TPI-Next 测试过程改进模型的背景、范围和目标。

问题：

以下关于 TPI-Next 测试成熟度矩阵的论述哪个是正确的？

选项：

A．针对关键域/成熟度级别的组合，对应的检查点显示在测试成熟度矩阵中
B．针对关键域/改进目标的组合，对应的检查点显示在测试成熟度矩阵中
C．针对改进目标/成熟度级别的组合，对应的检查点显示在测试成熟度矩阵中
D．针对关键域/成熟度级别的组合，对应的改进目标显示在测试成熟度矩阵中

解释：

A．正确。参考 *TPI-Next* 书本第 50 页。
B．不正确。改进目标在测试成熟度矩阵中不可见。

C．不正确。
D．不正确。
分值：1 分

44．CTAL-ATM_LO-5.6.1

TM-5.6.1(K2)总结 CTP 测试过程改进模型的背景、范围和目标。

问题：

下面哪个属于 CTP 测试过程改进达到目标的例子？

选项:

A．测试团队的缺陷检测有效性超过了行业平均值
B．测试团队的测试过程成熟度级别从 2 提升到 3
C．测试团队的测试过程成熟度级别，从控制级到高效级
D．测试团队经历了严格的测试过程评估

解释：

A．正确。CTP 参照行业平均数据，利用度量评估组织能力。
B．不正确。属于 TMMi。
C．不正确。属于 TPI-Next。
D．不正确。评估是 CTP 模型中的一部分，但它不是目标（除非是咨询师完成的）。
分值：1 分

45．CTAL-ATM_LO-5.7.1

TM-5.7.1(K2)总结 STEP 测试过程改进模型的背景、范围和目标。

问题：

你在国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

你的产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部的定制测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。你的公司已在公司级别实施了 CMMi 三级。

团队在赶上硬件开发进度方面存在问题。从长远来看，你认为 STEP 模型很适合解决上述问题。

下面哪个 STEP 基本前提最满足你的要求？

选项：

A．测试人员与开发人员一起工作
B．基于需求的测试策略
C．测试件设计领先软件设计
D．系统化地分析缺陷

解释：

A．正确。符合测试跟不上进度的问题。
B．不正确。B 选项本身是正确的，但是在该场景中没有提及需求的问题。
C．不正确。该场景中没有提及设计问题。
D．不正确。D 选项本身是正确的，但是已经完成该工作，因为缺陷已经分类。

分值：1 分

第6章 测试工具及自动化

本章学习目标如表 6-1 所示。

表 6-1 学习目标

编号	学习目标描述	级别
TM-6.2.1	描述选择开源工具时需要考虑的管理问题	K2
TM-6.2.2	描述决定定制工具时需要考虑的管理问题	K2
TM-6.2.3	对一个给定的情况做出评估，来制订选择工具的计划，包括风险、成本和收益	K4
TM-6.3.1	阐述一个工具的生命周期的不同阶段	K2
TM-6.4.1	阐述如何通过使用工具来改进度量的收集和评估	K2

本章相关术语如表 6-2 所示。

表 6-2 术语

英文	中文	说明
open-source tool	开源工具	源代码公开的软件工具。通常经互联网提供给用户使用，允许用户在许可范围内学习、修改、完善或分发
custom tool	定制工具	为用户或客户而特定开发的软件工具

6.1 简介

ISTBQ 基础级大纲中描述了各种类型的测试工具，不同的实际使用者，例如，测试经理、测试分析师等其采用的测试工具会有所不同。本章节主要阐述了测试经理在选择测试工具和自动化方面必须考虑的一些因素。

6.2 选择工具

测试过程中正确地使用测试工具，可以极大地提升测试效率和有效性。测试经理在选择测试工具时，必须考虑许多不同的问题。根据工具来源的不同，可以将测试工具分为商业工具、开源工具和定制工具。商业工具指的是直接从商业供应商处购买工具，有时候这是唯一可行的选择。而定制工具指的是为客户或用户专门开发的工具。本章节将会简单介绍开源工具和定制工具。

不管选择哪种类型的工具，测试经理都必须分析其中的收益和风险，并通过成本收益分析仔细研究该测试工具在预期使用年限内的所有成本，详细内容可以参考 6.2.3 节。另外，本章还会谈到选择测试工具的具体流程和注意点。

6.2.1　开源工具

测试过程中的几乎所有的测试活动都可以有开源工具的支撑，借助开源工具完全可以构造一个完整的测试工具解决方案，从组件测试到系统测试、从功能测试到性能测试、从测试用例管理到测试用例自动化，直至测试的管理平台和缺陷跟踪系统，开源测试工具能覆盖整个测试过程。

☆示例：开源测试工具

JUnit 是由 Erich Gamma 和 Kent Beck 完成的一个编写和运行自动化测试的框架。

Mantis 是一款基于 Web 的软件缺陷管理工具，配置和使用都很简单，适合中小型软件开发团队。

开源并不仅仅意味着开源的软件需要公开他们的源代码。根据开放源码促进会（Open Source Initiative，OSI）的解释，开放源码通过支持源代码的独立同行评审（Independent Peer Review）和快速发展演变提高了软件的可靠性和质量。要通过 OSI 认证，软件必须在获得许可证的情况下发布，该许可证可保证免费读取、重新发布、修改和使用该软件的权利。根据开放源码促进会 OSI 的定义，开放源码软件的发布必须遵从以下标准①。

1．免费重新发布

当软件是几个不同来源的程序集成后的软件发行版本中的一个组件时，许可证不能限制任何团体销售或分发该软件，并且不能向这样的销售或分发收取许可费和其他费用。

2．源代码

程序必须包含源代码，并且必须允许以源代码或已编译的形式发布。如程序在发布时未带源代码，则必须以一种非常公开的方式，在不超过合理重造成本的情况下，让人们获得源代码，例如，可以在不收取费用的情况下，放在网络上供人们下载。源代码无疑是编程人员最容易修改程序的形式。但是不允许故意混乱源代码，也不允许使用中间形式，例如：预处理器或转换器的输出。

3．衍生产品

许可证必须允许修改源代码产品和衍生产品，并且必须允许在与原始软件相同的授权情况下，发布修改之后的产品。

4．作者源代码的完整性

许可证可以禁止他人以修改过的形式发布源代码，只在该许可证基于修改程序的目的时，才允许随源代码发布“补丁文件”。该许可证必须明确允许发布根据修改过的源代码构

① 更详细的关于开源软件的介绍，参见 IBM 的网站（http://www.ibm.com/developerworks/cn/opensource/newto/）或 OSI 的网站（http://www.opensource.org/docs/definition.php）。

建的软件。许可证可能要求衍生产品必须附加不同于原始软件的名称或版本号。

5．不得歧视任何人或团体

许可证不得歧视任何人或任何团体。

6．不得歧视程序在任何领域内的使用

许可证不得禁止任何人在特定领域内使用某一程序，例如，不得禁止程序在商业上的应用，或者在基因研究上的使用。

7．许可证的发布

附加在程序上的权利必须应用于那些使用重新发布程序的人，无须通过其他人额外加以授权使用。

8．许可证不得专属于特定产品

附属于程序的权利不得仅限于作为特定软件发行版本的一部分程序。如果程序衍生自该发行版，并以获得该程序的授权的名义被使用或发布，则使用重新发布的该程序的其他所有人应该享有原始软件发行版本中所授予的那些权利。

9．许可证不得对其他软件加以限制

许可证不得对其他随已许可的软件一起发布的软件附加任何限制，例如，不得规定在相同媒体上发布的其他所有的程序接受该许可证的限制。

10．许可证必须是技术中立的

任何许可证规定都不可以基于任何单独技术或界面风格。

很多开源工具最初是为解决某个特定问题或针对单个问题而创建的，因此该工具不一定能实现与其类似商业工具的所有功能。因此在选择开源工具之前，需要对测试团队的实际需求进行周密的分析。

使用开源工具有一个很重要的特质，即开源工具本身的初始采购成本通常不高，因此测试团队直接投入的前期成本较少，但是带来的问题是可能没有任何正式可用的技术支持。然而，还是有些开源工具为用户提供了免费的后续跟踪，以进行非正式的技术支持。

使用开源工具除了初始成本低之外，其优点还包括：可以根据需要修改或者扩展工具，前提条件是测试团队具备修改工具的核心能力。通过修改开源工具，可以与其他测试工具集成在一起使用，或者以适应测试团队的实际需要。另外，测试团队可以组合多个开源工具以解决商业工具所不能解决的问题。当然，集成的工具越多，修改的内容越多，将导致工具复杂度和成本的不断提高。测试经理必须确保使用工具的目的是为了获得正的投资回报率，而不是为了尝试使用工具而采用开源工具。

测试经理在选择开源工具时，必须理解所选工具的许可协议组合。很多开源工具的协议属于 GNU（通用公共许可）的变化版本，规定软件分发的条件必须与收到软件的条件相同。当测试团队为了更好地支持测试而改变测试工具时，这些改动需要提供给在该工具许可下的所有外部用户。测试经理应该了解所在组织再次发布该软件而导致的可能法律后果。

假如测试团队负责测试的软件产品属于安全关键系统，在使用开源工具时可能会遇到问题。尽管开源工具具有扩展自由和前期成本低等特点，但是其在服务、集成度和功能等方面与商业工具有一些差距，特别是开源工具在质量和准确性等方面不能被证实，而商业工具通常可以证明其准确性和对特定任务的适用性（例如满足 DO-178B 标准）。因此，尽管开源工具不一定比商业工具逊色，但对开源工具的认证需要使用该工具的人去做，导致

使用开源工具的额外成本。

6.2.2　定制工具

由于可能存在独家的硬件平台、特殊的测试环境或者独特的工作流程，导致商业工具和开源工具无法满足组织的一些特定需求。或者不同项目的测试活动的特点各不相同，尽管很多商业软件能进行一定程度的定制，但是很多情况下仍然无法满足测试团队的需要。假如测试团队具备技术核心能力，测试经理可以考虑开发满足要求的定制工具。

自行开发的测试工具通常能有效地完成事先定义的任务，可以很好地满足测试团队的实际需求，并能在测试团队所要求的测试环境下高效地运行。定制工具可以开发成与其他工具交互的形式，并生成测试团队所需的特定格式的数据。同时，定制的工具还可以用于组织内的其他软件项目，需要注意的是其他软件项目在使用该定制工具之前，先评审使用该工具的目的、目标、收益和潜在缺点是很重要的。

测试经理在考虑开发定制工具时，还需要考虑面临的一些挑战。

首先，为了保证测试工具的质量，最好安排专门的测试工具开发人员进行开发。测试经理必须牢记，定制工具同样也是一款软件产品，正因为如此，它也会出现其他软件产品类似的开发问题。定制工具也需要进行设计和测试，以保证它们按照预期运行。因此必然会带来人力资源和时间成本等的额外要求。

其次，定制工具往往依赖于开发工具的人员，为了能让其他人维护测试工具，测试工具的开发和使用过程需要文档化。否则在开发工具的人离开项目后，定制的工具将由于无法维护而导致无人问津，甚至最终被弃用。随着时间的推移，定制工具的应用范围可能会扩展，超出最初创建工具的动机，而这也许会导致工具的质量问题，例如，误报的缺陷报告或创建了不准确的数据。定制工具的早期版本并不一定能带来效益，为了满足不断增加的需求需要对测试工具进行持续的扩展。

☆示例：iBAS 项目自动化测试工具开发

下面介绍 iBAS 项目自动化测试工具的开发情况。

由于宽带接入设备本身的结构复杂，且通信设备对于产品质量要求很高，所以对该设备的测试工作显得尤为复杂。在 iBAS 的集成和系统测试中，开发人员和测试人员缺乏方便的代码运行环境，一般都要依赖于机架进行测试。要想完成功能测试和非功能测试，需要一套完整的组网环境，测试时必须独占整个测试环境，造成其他测试无法进行，严重浪费了实验室的测试环境资源。同时，随着客户需求的增加和宽带业务的快速增长，目前的基本业务和补充业务数量非常多，在较短的时间内和有限的人力资源条件下测试如此多的业务非常困难。这是项目管理人员和测试人员都必须面对和不得不解决的问题。

总的来讲，目前 iBAS 的测试面对的难题主要有以下 5 个方面。

1. 设备用户接口的多样化

在实现用户业务之前，需要对设备进行相关的配置，但是 iBAS 提供的配置接口比较丰富，可以通过 Telnet、SNMP、CLI 和 TL1 等方式来配置。这就意味着测试需

要覆盖多种配置接口。在测试中，即使通过 Telnet 的方式能够对设备进行正确的配置，也并不意味着通过 SNMP、CLI 或 TL1 的方式也能正确地配置。

2. 设备硬件结构复杂

设备本身的硬件结构是由多块单板组成的，包括背板、系统管理板和高速接口板等，同时高速接口板又分为很多种不同的类型，例如，100M 电口、1000M 光口等。同样的测试用例可能需要在多个不同的硬件平台上运行。同时，针对不同的硬件或单板，测试用例中对部分返回信息的判断各不相同。这就要求软件测试自动化平台在设计的时候，需要考虑相同的测试用例如何在只需最少修改情况下对多块单板进行测试。

3. 业务组网复杂

用户业务方式也是多样的，用户可以通过 DHCP、PPPoE 等不同的方式接入网络。测试过程中不仅要用到被测试设备本身，还可能需要用到测试仪器、控制终端、交换机和各种服务器等。业务组网的复杂性不仅使得测试环境的搭建更加复杂，同时使得测试中出现问题时的定位也更加困难。

4. 需要专用的测试仪表

为了更好地模拟真实的用户环境，测试过程需要用到专用的测试仪表，例如，AX4000、Smartbits6000 等，同时由于设备应用的复杂性，涉及各种 TCP/IP 协议（例如 DHCP、IGMP 等），在测试过程中需要用测试仪表模拟这些协议的交互，而这些协议通过手工的方式构造很复杂。

5. 回归测试工作量巨大

iBAS 的需求很多且不稳定，产品开发周期较长，采用增量迭代开发的开发过程。这就意味着在整个产品的测试过程中，需要进行大量的回归测试。大量的回归测试不仅使得测试人员疲于奔命，同时也极大地影响了测试人员的工作积极性。

面对 iBAS 项目的这些测试难点，在测试中引入自动化将是必然的发展方向，自动化平台的引入可以减轻测试人员的负担，充分利用测试的各方面资源，加快软件版本的开发进度。通过自动化测试人员的努力，最终自行开发完成了软件测试自动化工具。该工具的架构如图 6-1 所示。

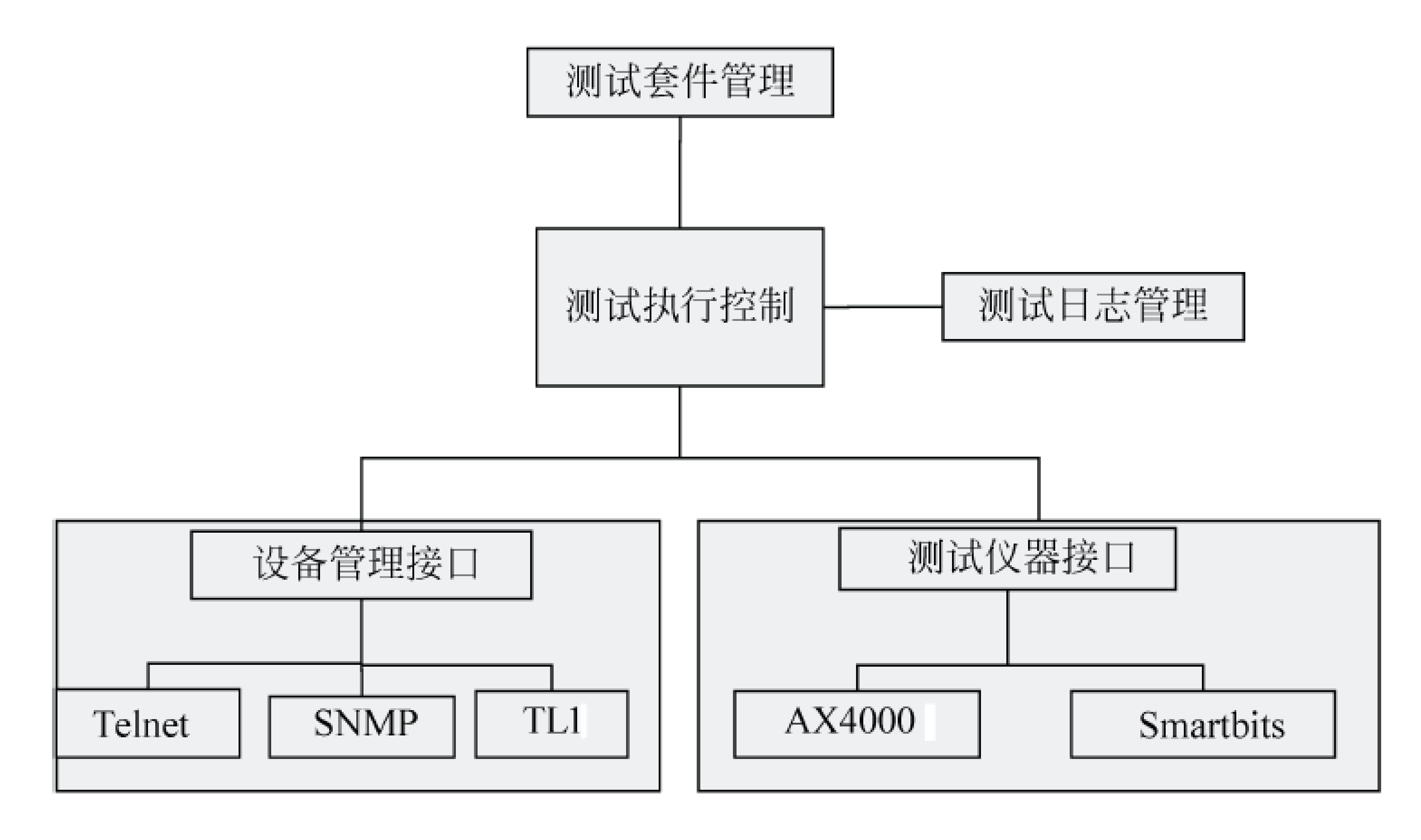


图 6-1 iBAS 自动化测试工具架构

6.2.3　投资回报率

测试经理在引入新的测试工具时，需要考虑选择、购买和维护工具的成本。另外，测试工具成本会随着硬件购买、工具升级以及员工培训而增加。根据测试工具的复杂度以及部署的工具数量，成本会迅速增长。与其他投资一样，测试工具的引入还需要考虑新工具开始显现成效的时间周期，即时间成本。例如，采用自动化测试执行工具之前，需要估计自动化测试相对手工测试所能节省的人力成本的可能性，可能需要的额外开发测试自动化脚本的工作量。通常情况下，测试脚本仅运行一次或有限几次都会使得成本大于收益，这也使多次重复执行的回归测试更适合采用自动化方式，其节约的成本会不断得到积累。

测试团队中引入测试工具，测试经理需要确保工具的引入可以为团队工作增加附加值，即为测试团队的工作带来正的投资回报，不断提高测试团队的效率和有效性。为了确保测试工具可以实现真正的持续收益，在购买或者开发工具之前，应该进行工具的成本-收益分析。

测试工具的成本-收益分析，需要考虑临时成本和重复成本，其中包括货币成本、资源成本、时间成本等，以及可能降低测试工具价值的风险。成本-收益分析还需要考虑使用工具所需的成本和不使用工具所需的成本之间进行对比。

临时成本，有时也称为一次性成本，主要包括以下成本。

（1）为了达到组织测试目标和目的，定义测试工具需求的成本；

（2）评价和选择正确测试工具和工具供应商的成本；

（3）采购、调整或开发测试工具的成本；

（4）测试工具的初始培训的成本；

（5）将测试工具与其他工具集成所需的成本；

（6）采购支持测试工具所需的硬件 / 软件的成本。

重复成本，有时也称为经常性成本，主要包括以下成本。

（1）持有测试工具的成本，包括：

① 测试工具的许可和支持费用；

② 测试工具后续的维护费用；

③ 由测试工具创建的工作产品的维护成本；

④ 持续的测试工具培训和辅导成本。

（2）将测试工具移植到不同环境所需的成本。

（3）调整测试工具以适应未来需求的成本。

（4）改进质量和过程确保已选测试工具最佳使用所需的成本。

假如仅考虑测试工具的使用成本，经常会遗漏一些重要的成本，例如，系统变更时维护、升级和完善测试脚本的成本等。除了使用成本，测试经理还应该考虑测试工具的机会成本，即花在采购、管理、培训以及使用该工具上的时间，可以花在其他的测试任务上。因此在测试工具上线前，可能需要更多的测试资源。

许多测试自动化的实施是基于已有的手工测试用例，在实施前应该仔细考虑测试用例是否适合自动化。而测试实践中往往缺少评估测试用例自动化中可能带来的实际收益。因

为执行自动化测试脚本的首个版本所需的成本，通常远高于手工执行测试用例的成本。随着时间的推移，测试自动化可以逐步提高测试执行的效率，以及显著改进测试覆盖率。下面是自动化测试用例挑选的案例。

☆示例：iBAS R1.0 项目的自动化测试用例挑选标准

iBAS R1.0 项目中，为了保证自动化的性价比，在挑选哪些测试用例进行自动化的时候有着严格的评判标准。在完成测试用例设计后，需要根据该标准对所有的测试用例进行筛选，以确定哪些测试用例需要进行自动化。该自动化测试用例挑选标准采用积分制，对每个测试用例按照该标准进行打分，根据最终的得分结果决定测试用例的自动化情况。

测试用例的打分标准如下。

从产品（域）的代码行数考虑，行数越多，越倾向于实现自动化测试。

1 分：产品（域）的代码低于 10 000 行。

2 分：产品（域）的代码介于 10 000 行和 30 000 行之间。

3 分：产品（域）的代码介于 30 000 行和 70 000 行之间。

4 分：产品（域）的代码高于 70 000 行。

系统的复杂性：系统越复杂也就越需要执行自动测试。

积分为 1～4，1 分表示系统复杂性最低，4 分表示系统复杂性最高。

按照测试失败对产品销售、使用和维护带来的风险进行评估。

1 分：测试失败，对产品销售、使用或维护没有任何威胁。

2 分：测试失败，对产品销售、使用或维护影响很小。

3 分：测试失败，对产品销售、使用或维护影响很大。

4 分：测试失败，对产品销售、使用和维护产生难以恢复的影响。

自动测试可行性。

1 分：测试本身难以自动化，并且测试平台要做改动。

2 分：测试本身难以自动化，但是测试平台不需要改动。

3 分：测试本身简单能自动化，但是测试平台需要改动。

4 分：测试本身简单能自动化，并且测试平台不需要改动。

测试手工执行，以分钟计算所需要的时间。

1 分：手工测试执行时间少于 5 分钟。

2 分：手工测试时间介于 5～15 分钟之间。

3 分：手工测试时间介于 15～30 分钟之间。

4 分：手工测试时间超过 30 分钟。

在后续项目使用的可能性。

1 分：机会很低。

2 分：机会中等。

3 分：机会很高。

测试的可维护性。

1 分：在后续版本中，需要很高的维护成本。

2 分：在后续版本中，需要的维护成本适中。

3 分：一旦完成，测试能维持稳定。

利用上面的评判标准对所有的测试用例进行评估，然后根据评估结果挑选要进行自动化的测试用例。通常情况下：

积分在 20 ~ 26 之间的测试用例，建议对它们进行自动化。

积分在 14 ~ 19 的测试用例，在时间允许的情况下，可以考虑进行自动化。

对于积分在 0 ~ 13 的测试用例，不建议进行自动化。

测试团队仅购买或租用测试工具，并不能保证成功地进行自动化测试。测试团队需要在测试工具上花费额外的工作量，例如，维护工作量，才能获得真正且持续的成效。尽管使用测试工具能给测试带来巨大的潜在收益和新的机会，但是同时必须考虑使用工具可能存在的风险。

使用工具的风险如下。

（1）组织的开发测试成熟度低，并未准备好使用测试工具；

（2）对工具存在不切实际的期望（包括对工具功能性和易用性抱有不切实际的期望）；

（3）低估首次引入工具所需的时间、成本和工作量（包括培训和获取外部的咨询）；

（4）低估从工具中获得较大和长久收益需要付出的时间和工作量（包括更改测试过程并不断改进工具使用方式的需要）；

（5）低估对测试工具生成的结果进行维护所需的工作量；

（6）对测试工具过分依赖（替代测试设计或者在一些更适合手工测试的地方强行使用测试工具）；

（7）不完备的或不正确的手工测试被自动化；

（8）当被测试软件改变时，测试件（例如脚本）难以维护，需要多次修改；

（9）测试活动中缺少测试人员的直接参与会降低缺陷发现率，因为只有自动的、程序化的测试在运行。

使用工具的潜在收益如下。

（1）减少重复性的工作（例如，执行回归测试、重新输入相同测试数据和按编程规范检查代码）；

（2）更好的一致性和可重复性（例如:，用工具执行测试、从需求导出测试）；

（3）客观的评估（例如：静态分析、覆盖率）；

（4）容易得到测试的相关信息（例如，关于测试进度的统计和图表、缺陷发生率和产品性能）；

（5）自动测试执行时间变得更易于预测；

（6）测试用例自动化有助于后期回归测试和确认测试更加快捷和安全地开展；

（7）提升测试人员和测试团队的技能水平；

（8）在增量-迭代开发过程中，自动化测试可以为每个增量-迭代提供更好的回归测试；

（9）能覆盖一些手工无法执行的测试类型（例如，性能测试和可靠性测试）。

应用测试工具需要综合考虑其临时成本、重复成本、风险和收益等因素，并基于此开展成本-收益评估。测试工具的应用必须是基于长期的规划。同时，测试团队会在测试过程

中用到各种测试工具，而团队获得的总的投资回报，应该是基于所有测试工具的使用。测试工具之间需要共享信息，相互协作。为了更好地使用测试工具，测试团队内构建一个长期而全面的测试策略是必需的。

6.2.4 选择流程

明确测试工具所要支持的测试任务后，测试经理接下来需要进行选择和评估测试工具的工作。选择和应用测试工具是个长期的投资，因此应该谨慎并且有计划地进行新工具的选择。测试工具可以应用一个项目的多个迭代，也可以适用于多个软件项目。在选择测试工具过程中，测试经理必须从以下几个角度考虑。

（1）业务层面，投资回报率必须是正的。为了提高测试工具投资回报率，组织应该确保工具之间的互操作性，例如，测试工具和其他工具之间能够交互操作。为了实现工具之间的互操作性，有必要改进开发测试过程与工具使用之间的衔接能力，这都需要组织在资源、时间和成本等方面的支持。

（2）项目层面，工具必须是有效的，例如，避免手工测试中类似数据输入时的录入错误。一般在实施测试自动化的项目中很难在项目初期就获得正的投资回报率，通常需要在较长时间后才能获取正收益，比如，几个版本发布后或维护测试时。测试经理需要从整个软件测试生命周期考虑测试工具的投资回报。

（3）对使用工具的人来说，该测试工具必须支持项目成员更加高效地完成自己的任务。在选择测试工具时，必须考虑团队成员掌握工具使用的学习曲线，确保他们能够在尽量小的压力下快速地学会使用工具。在初次引入测试工具时，需要对工具使用者进行培训和辅导。

为了确保考虑到测试工具的利益干系人的所有观点，测试经理需要为测试工具的引入创建规划路径。在 ISTQB 基础级大纲中已经讨论了选择测试工具的流程，包括：

（1）评估组织的成熟度；
（2）识别工具应用的需求；
（3）评估市场上各种工具；
（4）评估供应商或服务支持商（包括开源工具、定制工具）的能力；
（5）识别为使用工具需要进行训练和辅导的内部需求；
（6）考虑目前测试团队的测试自动化技能，评估培训需求；
（7）估算成本-收益（如在 6.2.3 节关于投资回报率的讨论）。

针对第（2）步中“识别工具应用的需求”，测试经理需要考虑下面的一些准则。

（1）与潜在测试对象进行交互的质量；
（2）测试人员对工具或方法的知识积累；
（3）集成到现有开发环境的便捷性；
（4）与使用的其他工具集成的可能性；
（5）工具要部署的系统平台；
（6）制造商的服务、可靠性和市场地位；
（7）工具的许可条件、价格、维护成本。

将上述准则以及将来可能进一步出现的准则汇编到列表中，并根据它们的相对重要性进行权衡，鉴别出绝对必要的准则和可选的准则，分别进行标记。

在评估供应商过程中，可以根据市场调研创建一份工具列表，在其中列出感兴趣的工具类型中的可用产品。产品信息可以从供应商那里获取，或从互联网收集。根据这些资料，邀请首选产品的供应商演示产品。通过演示，至少对该公司留下一些印象，并获取一些服务体系信息。最佳厂商将进入最终的评估过程，主要需要验证如下几点。

（1）该工具是否适用于测试对象和开发环境；

（2）促使工具进入最终评估的特性和质量特征是否能真正发挥作用；

（3）供应商的支持人员是否能够正确解答各种疑问（售前和售后）。

软件测试生命周期中的任何测试阶段，都会有合适的测试工具类型支持测试活动。以测试设计和自动化测试工具为例，选择测试工具时测试经理应当考虑测试工具是否能提供以下功能。

（1）分析能力，例如：

① 该测试工具是否能“理解”给定的输入？

② 测试工具是否能满足测试目的要求？

（2）设计能力，例如：

① 此工具是否能根据现有信息帮助设计测试件（例如，测试设计工具根据需求生成相应的测试用例）？

② 能否自动生成测试设计？

③ 能否以可维护和可使用的格式生成实际的测试件代码？

④ 能否自动生成必要的测试数据（例如，基于代码分析生成测试数据）？

（3）数据及测试选择能力，例如：

① 工具如何选择所需测试数据（例如使用哪组数据执行哪个测试用例）？

② 工具能否接受人工或自动输入的测试选择准则？

③ 工具能否根据已选的输入来决定如何过滤产品数据？

④ 工具能否根据覆盖准则来决定需要执行哪些测试（例如提供一组需求后，工具能否顺着可追溯性要求决定需要执行哪些测试用例）？

（4）执行能力，例如：

① 工具能否自动运行，还是需要人工干预？

② 工具如何停止和重启测试执行？

③ 工具能否“领会”关联事件（例如通过某测试用例发现的缺陷报告已处于关闭状态，测试管理软件是否能自动更新测试用例状态）？

（5）评估能力，例如：

① 工具如何判断它是否收到了合适的结果（例如工具将使用测试准则决定响应的正确性）？

② 工具具备什么样的错误恢复能力？

③ 工具能否提供充分的记录和报告？

6.3 工具生命周期

与其他测试活动一样，测试工具也有它的生命周期。工具生命周期由不同的 4 个阶段组成，分别是：获取阶段、支持与维护阶段、演变阶段和退役阶段。为了更好地发挥测试工具在测试任务中的作用，测试经理必须对这 4 个阶段进行有效的管理。

1．获取阶段

测试工具的获取，必须按照 6.2.3 节进行成本-收益分析，并基于正确的选择流程进行工具选择。

测试经理决定在测试团队中引入某测试工具之后，应当指定合适的测试分析师或者技术测试分析师负责管理该工具。测试工具负责人需要确定使用工具的时机和方式，例如，如何对测试工具生成的测试件进行储存，定义文件和测试的命名规则、创建库和定义模块化测试套件等。

测试工具的使用、管理、存储和维护等标准应该在使用工具前确定，而不是等到要用工具再临时去做决定。测试工具前期准备工作是否充分，最终也会影响到工具投资回报率。高效使用测试工具的一个手段是尽早对工具的使用者进行有效的指导和培训。

2．支持与维护阶段

测试团队获得测试工具之后，需要对工具进行持续的支持和维护。通常维护测试工具的责任会落在工具管理人身上，或者分配到专门的测试工具团队。假如该测试工具需要与其他工具一起运行使用，这时需要考虑工具之间的数据交换和协作过程。另外，测试工具的支持与维护过程中，必须考虑测试工具的输出工作产品，以及它们的备份和恢复能力。

3．演变阶段

随着时间的推移，测试环境、业务需求或者工具供应商问题等原因，都可能要求对工具本身或者其用途进行大的变更，因此必须在工具使用中考虑工具转换的能力。例如，测试工具供应商对工具功能的增强，可能会导致与其他工具之间的集成出现问题；或者由于被测对象的业务原因导致的环境变动，也可能需要对测试工具进行变更。

测试工具的运行环境越复杂，工具本身的变更或者业务方面的变更，都可能导致工具无法正常使用。根据测试工具在测试过程中扮演的角色，测试经理需要确保工具在出现问题时，能保证工具提供服务的连续性。

4．退役阶段

随着软件产品的逐步成熟和最终完成，相应的测试工具也将逐步退出。在测试工具退役阶段，该工具所支持的功能需要有其他工具替代，该工具输出的数据需要进行保存和归档。简单地讲，当使用新工具的收益已经超越了旧工具的成本和风险时，旧工具就可以退役了。

测试经理的主要职责是，确保在测试工具的整个生命周期中，工具能为测试团队顺利运作并提供持续的服务。

6.4 工具度量

测试经理在测试过程中需要设计、收集和分析各种度量数据，而这些数据很多都是来自测试分析师或者技术测试分析师使用的测试工具。不同工具可以收集不同的实时数据，同时可以减少测试团队收集数据的工作量。测试经理可以基于这些度量数据进行管理测试工作，例如，监控测试执行进度。

不同的测试工具，其收集的数据类型是不同的。例如：

（1）测试管理工具贯穿于整个软件开发生命周期，可以为测试经理提供多种度量数据。从需求到测试用例再到自动化测试脚本的可追溯性度量数据，以评估相应的测试覆盖率。在需要时可获取当前有效的测试用例、计划的测试用例，以及当前测试用例执行的状态，例如，通过、失败、被阻塞等。

（2）缺陷管理工具可提供缺陷相关的丰富信息，例如，缺陷状态、严重程度和优先级、缺陷在不同模块中的分布等。其他常用的数据还包括缺陷的引入阶段、缺陷发现阶段、缺陷遗漏率等，这些缺陷度量数据可以有效地帮助测试经理推动过程改进。

（3）静态分析工具可为发现和报告可维护性问题提供帮助，例如，是否和定义的编程规范要求一致、过于复杂的代码、引用没有定义的变量等。

（4）性能工具可提供在系统可扩展性方面有价值的信息。

（5）覆盖率工具可以帮助测试经理了解系统实际完成测试的比例。

测试工具需要提供什么类型的度量数据，应该在工具选择时就确定数据报告的需求。并在测试工具实施过程中得到合适的实现，从而确保通过测试工具跟踪的信息能够以利益干系人可以理解的方式进行汇报。

小结

测试过程中不同的测试活动，需要不同测试工具的支持。根据工具的来源不同，可以分为开源工具和定制工具。有的工具只支持一种测试活动，而有的工具可以支持多种测试活动。

不管是开源工具还是定制工具，在选择过程中都需要评估各自的特点、潜在的收益和风险，以及投资回报率。仅购买或租用工具并不能保证成功使用工具。任何类型的工具都需要额外的工作量才能获得真正且持续的成效。工具的支持能给测试带来巨大的潜在（可能）收益和新的机会，但是必须考虑到工具的使用同样存在风险。

与测试过程中的测试活动一样，测试工具同样具有生命周期：获取阶段、支持与维护阶段、演变阶段和退役阶段。测试经理需要对每个阶段的活动进行有效的管理，确保在测试工具的整个生命周期中，工具能为测试团队顺利运作并提供持续的服务。

测试工具可以在测试过程中提供各种度量数据。测试经理根据不同的管理目标，例如，监控测试执行进度，通过工具收集和分析不同维度的度量数据，对测试管理工作进行有效的管理。

模拟题

46．CTAL-ATM _LO-6.2.1

TM-6.2.1(K2)描述选择开源工具时需要考虑的管理问题。

问题：

你在某国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年工作时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部定制的测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时管理层安排工作量对个人效率的度量开展度量。公司已在公司级别实施了 CMMi 三级。

团队在紧跟硬件开发进度方面存在问题。公司原来开发了内部定制的测试自动化工具，经常需要针对被测系统构建接口驱动测试脚本，以完成所有通信标准的需求。

维护内部定制工具的成本越来越高。

测试自动化专家推荐了一款试用版本的开源工具。在做出决定之前你需要考虑几个问题。下面哪个问题是**不适合**的？

选项：

A．开源工具很难修改

B．需要理解许可证条款

C．需要考虑通信标准的一致性

D．开源工具是针对特定目的而开发的

解释：

A．正确。开源工具能够被修改，你必须具备这个能力。在前期开发过定制工具，因此不用担心很难适应的问题。

B．不正确。B 选项是应该关注的，因为你需要修改工具，可能需要将修改提交到开源社区，这依赖于许可证条款。

C．不正确。C 选项是应该关注的，因为需要通信标准的一致性。

D．不正确。D 选项是应该关注的，因为你有自己特定的目的（这也是为什么最初开发定制工具）。你的最初目的是否和需求差距太大？

分值：1 分

47．CTAL-ATM_LO-6.2.2

TM-6.2.2(K2)描述决定定制工具的管理问题。

问题：

你在某国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年工作时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部定制的测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。公司已在公司级别实施了 CMMi 三级。

团队在赶上硬件开发进度方面存在问题。

公司原先计划开发一款内部定制测试工具，原因之一是公司独特的硬件架构。测试工具的维护被证明过于花费时间。

你在考虑当前业务团队定制的工具是否还有用，考虑是否可以采用其他选项，例如，开源工具。下面哪个选项最好地支持采用定制化测试工具？

选项：

A．硬件组件经常会有大量的变动，因此测试工具的频繁变动也是需要的

B．公司必须遵循通信标准

C．公司有很多开发人员，有能力开发定制工具，因此需要利用这些技能

D. 工具易学易用

解释：

A. 正确：公司的硬件业务团队经常变动硬件（每 6 个月）。
B. 不正确：B 选项建议使用现成的工具，但是定制化工具也必须满足标准一致性，尽管可能需要更多的工作量。
C. 不正确。这是前提条件，而不是原因。
D. 不正确：这使得保留定制化工具有吸引力，但不能对使用和维护工具需要花费大量时间做出解释，题干没有提到易于使用。

分值：2 分

48. CTAL-ATM _LO-6.2.3

TM-6.2.3(K4) 对一个给定的情况做出评估，制定选择工具的计划，包括风险、成本和收益。

问题：

你在某国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年工作时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部定制的测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。公司已在公司级别实施了 CMMi 三级。

团队在赶上硬件开发进度方面存在问题。你听说在公司内部，另一个类似软件产品线的项目在使用开源工具进行他们的测试自动化。他们利用该工具将大约 50%的测试用例进行了自动化，而剩余的通过软件界面以手工方式测试。

现在要求你汇报是否有可能在产品线中也选择该工具。你主要关注什么？

选择最合适的三个选项。

选项：

A．你需要花费多少时间来重新编写已有的测试用例
B．是否可能通过手工方式执行一部分测试用例
C．通过使用新的工具，测试人员是否可以完成更多的工作
D．测试人员是否都能学会该新工具
E．该开源工具的后续支持如何
F．该新的工具是否易用
G．该工具是否存在安全性问题

解释：

A．正确。你应该考虑已有的大量测试用例，同时也应该关心投资回报率。
B．正确。你应该考虑该工具无法全部满足技术要求的可能性，即使它在某些方面很有效率。
C．正确。涉及整体的投资回报率 ROI，以及在项目截止时间之前能否快速完成的能力。
D．不正确。尽管通常是需要检查的一个点，但不是你主要关注的地方。
E．不正确。尽管通常是需要检查的一个点，但不是你主要关注的地方。
F．不正确。尽管通常是需要检查的一个点，但不是你主要关注的地方。
分值：2 分

49．CTAL-ATM_LO-6.3.1

TM-6.3.1(K2) 阐述一个工具的生命周期的不同阶段。

问题：

你在某国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年工作时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部定制的测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷的数目和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。公司已在公司级别实施了 CMMi 三级。

团队在紧跟硬件开发进度方面存在问题。

你听说在公司内部，另一个类似软件产品线的项目在使用开源工具进行他们的测试自动化。他们利用该工具将大约 50%的测试用例进行了自动化，而剩余的通过软件界面以手工方式测试。

假如你选择同一款开源工具，在当前内部定制工具退役之前，下面哪个活动是应该首先考虑的，以尽可能快地使之产生价值？

答案选项：

A. 回归测试脚本必须从内部定制工具转换到新的工具上

B. 必须继续维护内部定制的工具，并转换到新的环境

C. 内部定制工具的备份和恢复功能必须继续维护

D. 所有内部定制工具覆盖的测试脚本，都应该转换到新的工具上

解释：

A. 正确。选项 A 是有关工具退役的正确答案，因为有许多脚本已经存在，而回归测试脚本是最常被使用的。

B. 不正确。内部定制工具将会退役，因此该活动并不是必需的。

C. 不正确。内部定制工具将会退役，因此该活动并不是必需的。

D. 不正确。即使你希望这样尝试，转换所有的测试脚本也是不现实的，假如你能管理回归测试脚本已经很好了。

分值：1 分

50．CTAL-ATM_LO-6.4.1

TM-6.4.1(K2)阐述如何通过使用工具来改进度量的收集和评估。

问题：

你在某国际大公司工作，该公司研发硬件与软件集成的通信网络产品。硬件和软件开发由两个独立的业务团队完成。你是网络路由器软件产品线的测试经理。

产品线有一个长期的传统就是采用增量产品生命周期来开发高度集成的产品。硬件业务团队每 6 个月生成一个新的版本。软件产品线目标是针对每个新的硬件版本准备新的软件版本。软件以两个月一个增量的方式进行开发。

业务部门的进度在设计阶段就进行了同步。

测试团队由 15 个测试人员组成，他们在该公司至少都有两年时间，大部分都是有更长的工作时间。经验丰富的测试人员利用公司内部定制的测试脚本开发新的测试用例。测试团队的其他测试人员负责执行新的测试用例和回归测试用例集。

公司的管理层要求提供月度的进度报告，列出发现的严重缺陷数和测试执行的状态。同时提供工作量进行所有业务单元个人效率的度量。公司已在公司级别实施了 CMMi 三级。

团队在紧跟硬件开发进度方面存在问题。

通过进一步分析开发进度方面的问题后，初步认为没有时间充分覆盖新版本的新需求。

你在考虑如何测量功能测试脚本的覆盖率，以帮助在发布时间之前完成新需求的覆盖。下面哪个是最佳的选项？

选项：

A. 在测试管理工具中检查从测试脚本到测试需求的可追溯性

B. 在测试管理工具中收集每个增量中执行的测试脚本数量

C. 监控系统的性能，从而可以调整它的可扩展性

D. 作为测试脚本的一个属性，测量测试脚本开发所花的小时数

解释：

A. 正确。该度量可以自动告知是否充分覆盖了需求，从而可以判断是否满足版本发布条件，同时判断是否在某些区域覆盖率过高。

B. 不正确。该论述本身不提供任何信息，所有的测试可能来自同一个功能区域。但该信息可以让测试执行测量变得更容易一些。

C. 不正确。关注在性能测试工具上。

D. 不正确。这可能有助于更容易收集小时数，但它本身不提供有用的信息，在该场景中时间是面临的问题，因此可能会错误选择该选项。

分值：1 分

第7章 人员技能——团队构成

本章学习目标如表 7-1 所示。

表 7-1 学习目标

编　号	学习目标描述	级　别
TM-7.2.1	使用技能评估数据表，分析团队成员在使用软件系统、领域及业务知识、系统开发领域、软件测试及人际交往技能方面的强项和弱项	K4
TM-7.2.2	分析一次给定的团队技能评估，制订一份培训及技能培养计划	K4
TM-7.3.1	针对某一给定的情况，讨论领导测试团队所必须具备的硬技能和软技能	K2
TM-7.4.1	说明独立测试的可选做法	K2
TM-7.5.1	列举对测试人员动机产生积极和消极影响的因素	K2
TM-7.6.1	说明哪些因素会影响测试团队内的沟通，哪些因素会影响测试团队与干系人之间的沟通	K2

相关术语如表 7-2 所示。

表 7-2 术语

英　文	中　文	说　明
independence of testing	测试独立性	职责分离，有助于客观地进行测试[DO-178b]

7.1 简介

人是整个测试过程中最重要的组成部分，测试过程中的任何测试活动和任务最终都需要测试人员的参与完成，无论技术、工具或方法多么完美，最终都离不开人，因此测试经理必须意识到测试团队个人技能的重要性。测试经理所招聘和维护的测试团队成员，应该能让团队成员各有所长，能力方面互补。

随着测试项目的变化，对测试团队技能的要求也会随着时间而改变，因此提供适当的培训和成长机会也很重要，使得测试团队保持最佳的技能水平。对于测试经理而言，除了测试团队所需的技能之外，还需要具备一系列的软技能，使其在高压力和快节奏环境下也能有效地履行测试管理的职责。

本章首先讨论个人技能，然后着眼于如何评估测试技能，如何填补技能差距以创造内部凝聚力强大的测试团队，并在团队内开展有效的团队协作、激励以及沟通。

7.2 个人技能

个人的软件测试能力可以通过经验或者在不同工作领域的实践中获得。下面的每一条对增加和改进测试人员的基础知识都有帮助。

（1）软件系统的使用；

（2）领域或业务的知识；

（3）积极参与软件开发过程不同阶段的各种活动，包括分析、开发和技术支持等；

（4）积极参与软件测试过程中的各项活动；

（5）人际交往能力。

软件系统的用户非常了解系统中与用户有关的问题。例如，如何操作系统、失效对系统产生的最大影响以及期望的系统行为。具有专业知识的用户能够识别最重要的业务，以及这些业务如何影响系统的整体功能。测试人员需要了解用户是如何使用软件系统的，具备这些知识有助于测试人员设定测试活动优先级、有效设计测试用例和构造测试数据。

被测对象的领域或业务的知识是测试人员开展具体测试活动的基础。检查测试对象是否实现了系统的基本功能是测试的关注点之一，而这需要测试人员深入了解和理解测试对象要实现的功能或者服务，例如，iBAS 系统中的 IGMP 功能，作为测试人员，必须了解 IGMP 是如何工作的，同时需要深入了解 IGMP 相关的标准（RFC 1112 等）。不同的被测系统或者产品，其相关领域或者业务的知识要求是不一样的，例如，通信行业的软件系统和银行的网上交易系统，其功能和要求符合的标准是不一样的。

软件开发过程的知识（需求分析、设计和编码等）有助于深入了解错误如何被引入、错误发生的位置和如何预防它们的发生。软件产品相关的技术支持的经验有助于理解用户的体验、期望和易用性的需求。软件开发方面的经验也很重要，因为高端自动化测试工具的使用要求测试人员具有编程和设计方面的专业知识。

积极参与测试过程中的各项活动，有助于了解每个阶段需要做什么，以及阶段输出的工作产品。测试过程中的特定的测试技能包括分析规格说明、参与风险分析、设计测试用例以及运行测试和记录结果等。对于测试经理而言，他的项目管理知识、技能和经验是非常重要的，因为测试管理就像运行一个项目，需要制订计划、跟踪进度和向利益干系人报告测试状态等。在没有项目经理的情况下，测试经理可能承担起测试经理及项目经理两个角色的职责，尤其是在项目的后期。

人际交往能力（例如：批评与被批评、影响力和谈判能力）在测试中的地位也很重要。技术上能够胜任的测试人员并不一定能在测试工作中取得成功，因为测试人员除了技能方面的硬实力之外，还需要具备足够的软技能，例如，沟通技巧。除了与他人一起有效地工作外，专业的测试人员还必须条理清晰，关注细节，拥有良好的书面和口头表达能力。

理想的测试团队应该由不同技能和经验水平的测试人员构成，团队成员之间能够乐于相互帮助和学习。在不同测试项目环境和测试级别下，对测试人员的技能要求是不一样的，例如，API 测试中编程等专业技能可能会比业务领域知识更有价值；而在黑盒测试中，业务领域专业知识也许最有价值。切记测试环境和项目并不是一成不变的，因此对测试团队

的技能要求也是不断变化的。

测试经理在招聘测试人员时应该处于长远的考虑，而不仅是为了完成某个测试项目。测试经理不太可能招聘到完美的测试团队成员，即使当前团队成员对于目前的测试项目是适合的，但对于下个测试项目也许就不是完美的。因此，测试经理应该聘请聪明、好奇心强、适应能力强、乐于工作、善于学习，并能作为团队成员的一分子与其他成员协调工作的人员。当测试经理愿意为测试人员投入资源，并建立持续的学习环境时，可以不断激励并提高测试团队成员的技能和知识，并为下次的测试项目做好准备。

除了尽量招聘到合适的测试人员，测试经理也可以通过平衡团队中每个人的强项和弱项建立强大的测试团队。这时就需要测试经理创建测试技能评估数据表，以评估每个团队成员的强项和弱项。其基本思路是：首先，测试经理列出测试工作和岗位所需的重要技能；其次按照给定的打分机制（例如，从 1 分到 5 分，其中 5 分为该技能最高的能力等级），对团队中的每个成员个人能力进行评估；然后，根据评估结果判断团队中个人的强弱项，并基于这些信息制定个人或者团队的培训计划。测试经理可以为每个人创建特定领域提升能力的绩效目标，并确定评估个人技能的具体标准。

根据评估得到的测试团队个人技能强弱项，测试经理从影响团队有效性和效率的最可能弱项着手，以决定如何解决这些薄弱环节。主要的手段如下几项。

（1）培训，例如安排人员参加外部培训课程、内部培训、定制培训、在线培训课程等。

（2）自学，例如通过书籍、网络讨论、互联网资源等进行自学。

（3）交叉培训，例如，安排想要学习某技能的人员，与另一已经具备该技能的人员一起工作，在工作中学习该技能。或安排擅长该领域专业知识的专家进行面对面的交流。辅导也可以看成是交叉培训的一种类型。

除了解决测试团队中的薄弱环节外，测试经理还应该借助技能评估中发现的强项，帮助制定培训和技能发展计划。更多关于测试团队发展计划，可以参考 McKay07。

7.2.1 角色和职责

测试团队的构成和其规模没有多大关系，假如测试团队只有一个人，那么这个人需要扮演不同的角色。一般来说，完整的测试团队应该具有下面这些角色。

（1）测试经理：主要负责测试计划和测试控制，具备软件测试、质量管理、项目管理和人员管理等领域的知识和经验。

（2）测试设计人员：例如，测试分析师、技术测试分析师，他们是测试方法和测试规格说明方面的专家，具备测试设计、测试分析以及软件工程等领域的知识和经验。

（3）测试自动化人员：测试自动化专家，具备测试基础知识、编程经验以及丰富的测试工具和脚本语言知识。利用项目中提供的测试工具，按需要进行测试的自动化。

（4）测试环境管理员（实验室管理员）：安装和操作测试环境方面的专家，具备系统管理员知识。建立和支持测试环境，需要经常与系统管理员和网络管理员进行协调。

（5）测试执行人员：执行测试和事件报告方面的专家，具备 IT 基础知识、测试基础知识，能应用测试工具，熟悉被测试对象。测试员也常常作为以上提及的除测试经理外所有角色的通用名称。

优秀的测试团队需要对不同的角色明确定义不同的角色职责。明确的职责是高效的前提。下面一一介绍从测试经理开始到测试执行人员的职责。

1．测试经理

测试经理的职责如下。

（1）制定或评审组织内部的测试方针；

（2）选择合适的测试策略和方法，并引入或改善与测试相关的过程（缺陷管理、风险管理、配置管理等），以便能够管理和控制变更，以及重现缺陷和测试活动；

（3）作为整个项目的测试方面的代表（代言）；

（4）与项目经理以及其他相关人员共同制订测试计划；

（5）获取测试资源；

（6）发起和监视测试工作，包括各个测试级别相关的测试设计、实现和执行；

（7）根据测试结果和测试进度（例如：状态报告中的记录）及时调整测试计划；

（8）对测试件进行配置管理，保证测试件的可追溯性；

（9）利用合适的度量标准估算测试进度，评估测试和产品的质量；

（10）选择和引入合适的测试工具，并为测试人员组织必要的培训；

（11）确定测试环境和测试自动化的类型和范围；

（12）根据测试过程中收集的信息编写测试报告。

2．测试设计人员（测试分析师/技术测试分析师）

测试设计人员的职责如下。

（1）分析、评审和评估用户需求、规格说明、设计和模型等内容的可测试性，并设计测试用例；

（2）选择自动化测试用例；

（3）准备和获取测试数据。

3．自动化测试人员

自动化测试人员的职责如下。

（1）负责自动化测试框架的设计和搭建；

（2）负责搭建自动化测试环境；

（3）自动化测试需求分析，制订自动化测试计划；

（4）开发自动化测试用例；

（5）对自动化测试执行情况进行分析，完善自动化测试框架。

4．测试环境管理员

测试环境管理员的职责如下。

（1）负责测试环境所需的网络规划和建设，维护网络的正常运行；

（2）建立、设置和维护测试环境所需的应用服务器或软件平台；

（3）对实验室的硬件、软件资源进行登记、分配和管理；

（4）申请所需要的硬件和软件资源，协助有关部门进行采购和验收；

（5）对使用实验室的硬件、软件资源的使用权限进行设计、设置，保证其安全保密性；

（6）安装新的测试平台和被测试系统；

（7）优化测试环境，提高测试环境中网络、服务器和其他设备运行的性能。

5．测试执行人员

测试执行人员的职责如下。

（1）评审测试规格说明；

（2）建立测试环境（通常需要与测试环境管理员相互配合，共同完成）；

（3）执行各个级别的测试，记录测试日志，评估测试结果，记录实际结果和期望结果之间的偏差；

（4）根据要求使用测试管理工具和测试监控工具；

（5）执行自动化测试（可能需要开发人员和测试自动化人员的支持）。

根据项目或产品的测试级别及可能存在的测试风险，需要由不同的人充当不同的测试角色，同时保持一定的独立性。不同的测试级别，测试人员的组成可能是不一样的，例如，在组件和集成测试级别中，测试人员可能是开发人员；进行验收测试的测试人员一般是业务方面的专家或用户；而进行操作性验收测试的一般是将来使用软件的操作者。

7.2.2 软技能

软技能指那些“不易看见的技能”，是一个人“激发自己潜能和通过赢得他人认可和合作放大自己的资源，以获得超越自身独立能力的更大成功的技能”的总和。

软技能其实是情商 EQ（Emotional Intelligency Quotient）的社会学术语，它由一系列能够反映个人特质的要素组成，这些要素包括一个人的人格特质、社交能力、沟通能力、语言能力、个人行为习惯、待人友善、积极乐观等。软技能与硬技能（那些作为工作硬性要求并能够部分反映一个人的智商能力）是互补的。软技能（情商）可在一个组织的成功中扮演非常重要的角色。总的来说，软技能一方面体现在工作和生活中解决问题、处理问题的能力；另一方面体现在通过赢得他人的认可和合作，放大本人资源以获取更大成功的能力[①]。

测试人员在软件开发生命周期中，除了独立完成测试任务以外，还需要和项目的不同利益干系人进行合作，包括项目经理、开发人员或者用户等。测试人员需要向项目经理/测试经理反馈测试进度、产品质量等信息，同时还需要从项目经理/测试经理处获得项目的进展和状态，例如，项目内容或进度的变更。在测试过程中，无论是提交缺陷还是文档评审，测试人员都离不开和开发人员的合作和沟通。测试人员还可能需要从客户那里了解用户是如何使用产品的，或者因为产品的质量问题，从用户那里得到反馈甚至抱怨。因此，在复杂的测试工作环境中，测试人员除了必备专业知识和测试技能以外，还需要具备一定的软技能。下面介绍测试人员在测试过程中需要具备的主要软技能。

1．怀疑精神

发现缺陷是测试过程中的主要目的之一，因此，测试人员对被测产品要有怀疑精神，即怀疑测试对象存在缺陷或者是无法正常工作的。很难想象，缺乏怀疑精神的测试人员能够有效地发现软件产品中的缺陷，从而达到尽量多地发现缺陷这样的目标。在静态测试过程中（例如：评审），无论是系统的需求规格说明还是设计规格说明，测试人员都应该以

① 关于软技能的描述，来自 WiKi 中关于“Soft Skills”的介绍：http://en.wikipedia.org/wiki/Soft_skills。

怀疑的态度去对它们进行分析和评估；在动态测试过程中，也同样需要怀疑精神，这样才能更有效地发现缺陷。

怀疑精神不仅适用于开发活动的交付物上，同样也适用于测试团队的交付物上，例如，对于测试团队设计的测试用例，也不能盲目相信它们不会存在错误和缺陷，这些测试用例同样需要经过项目利益干系人的评审，例如，开发人员或客户；在测试执行的时候，发现实际结果和预期结果不一致时，首先需要确认测试用例是否正确，同时确保测试执行的过程、数据、操作等没有问题，然后确认是否是测试对象没有实现规格说明中要求的功能或者不一致，这都需要测试人员具有怀疑精神。

2．好奇心

好奇心是人们希望自己能知道或了解更多事物的不满足心态。好奇心不仅是要对某一事物感到疑惑和好奇，还要继续思索以求明白事情的真相，所以好奇心是创造的出发点、动机和推动力，也是产生无穷毅力和耐心的源泉。测试人员尤其是参与集成测试和系统测试的人员需要了解很多知识，他们不仅要掌握软件系统的相关功能模块，甚至是整个系统的需求和设计，同时还要从用户的角度思考系统的使用方式。如果测试人员只关注很小范围内的信息或知识，那么他不会成为一名优秀的测试人员。测试人员对整个产品和用户环境要充满好奇心，才能驱使自己不断地学习和进步，掌握更全面的知识。

另外，除了怀疑精神，好奇心也是测试人员找到更多缺陷的基础。当测试人员在测试软件系统的过程中，碰到了一个异常的系统表现行为，对于有好奇心的测试人员而言，他会深入研究为什么会出现异常行为，并通过不断地分析和探索，找到其中的原因，例如，由于软件产品中存在的隐秘缺陷，在偶然触发的情况下导致了这样的异常行为。而没有好奇心的测试人员，可能就会遗漏了这样的缺陷。

3．创新能力

创新能力能够根据已经存在的基础事物来创造一些新的特别的元素。创新能力可以通过一些好的实践来进行提高。在日常工作中，对问题的思考角度能够影响一个人的创新能力，当出现问题的时候，要尝试从不同的角度来分析问题，这样才能创造性地解决问题。还有一些好的习惯都有利于提高创新能力。当人处于比较放松的状态下、全面阅读尽可能多的知识、认为每个问题都是特别的、推迟做出决定和对新奇的想法保持开放的态度都有助于创新能力的提高。

创新同时也意味着要打破常规，避免重复性的思考。想要获得更好的创新能力，就不能害怕尝试新鲜事物，也不要对一个想法迅速做出肯定或否定的判断。从测试计划一直到测试结束活动，整个测试过程将会遇到各种各样的问题，这个时候就需要测试人员充分应用自己的创新能力，创造性地解决这些问题，以帮助项目获得更大的成功。

4．分析能力

随着软件系统和产品功能的不断增强，软件产品变得越来越复杂，这就使得测试活动也变得越来越困难，因此，对测试人员的分析能力要求也越来越高。测试过程是一个不断计划、分析和评估的过程，测试人员没有良好的分析能力，就无法高质量地完成测试任务，例如，测试计划和控制阶段，测试团队需要分析测试的范围、测试采用的技术和方法、测试的工作量、测试的风险等，根据测试过程中得到的信息和状态，分析计划和实际进度之间的偏差，调整测试资源、测试优先级和测试计划等；在测试分析和设计阶段，测试人员

需要不断学习和了解被测试对象的测试依据文档（例如需求规格说明），对被测试对象进行详细分析，确定如何采用测试计划中定义的测试技术和方法来设计相应的测试用例；在测试实现和执行阶段，根据测试执行过程中发现的失效，分析其失效的影响，并判断失效的严重程度和优先级，从而不断地调整测试资源的分配和测试任务的优先级。

5．耐心

软件开发一般被看作是建设性的活动，而软件测试常常在一定程度上被看成是破坏性的活动。但是软件测试的这种破坏性的活动并不是每次都能“破坏”成功的。在测试过程中经常出现这样的现象：尽管运行了很多的测试用例，但是没有在测试对象中发现任何的失效。这个时候，就要求测试人员具有足够的耐心。开发人员的重要输出是软件产品的代码，软件代码的数量会随着开发的深入持续不断地增长，这将极大地激发开发人员的成就感。但是测试人员不同，测试人员就像猎人一样，要经过漫长的“狩猎”过程才可能发现一个“猎物”。即使测试对象出现了异常行为，也需要测试人员经过耐心的分析和研究，才可能最终确定是否是缺陷。通常来说，测试对象的异常现象经常隐藏在大量的系统正常行为之中。因此，需要测试人员耐心检查各种数据、操作和系统表现行为，才能够发现其中的异常。

6．沟通技巧

测试并不是独立存在的，而是软件开发生命周期中的重要组成部分。实际的软件项目通常是由各种不同角色的人合作完成的，这样就难免会遇到需要和他人合作完成一项任务的问题，这时沟通就很关键。尤其是在当今全球化程度越来越高的情况下，很多项目由位于世界各地的团队共同完成，沟通就更加重要。沟通的方式有很多，可以是面对面的交流、电话或 E-mail，也可以通过文档和缺陷报告来交流信息。测试人员在整个项目开发过程中可能需要和项目经理、开发人员或用户等多种角色进行交流，良好的沟通能力有利于测试活动的开展。关于沟通技巧的更详细的内容，参见 7.6 节。

7．团队精神

随着社会的发展，社会的分工越来越细化，这同样适用于软件开发和软件测试，软件开发生命周期中的任务通常都需要通过团队来完成。团队并不是一群人的机械组合，真正的团队应该有共同的目标，其成员之间的行为相互依存和影响，并且团队成员之间能够很好地合作，从而追求集体的成功。团队是一个有机整体，团队成员除了具有独立完成工作的能力之外，同时还需要具有与他人合作共同完成工作的能力。团队的绩效源于团队成员个人的贡献，同时永远大于单个团队成员贡献的总和。团队精神是大局意识、协作精神和服务精神的集中体现。团队精神要求团队成员有统一的奋斗目标或价值观，而且需要相互信赖，需要正确而统一的企业文化理念的传递和灌输。团队精神强调的是组织内部成员间的合作态度，为了一个统一的目标，成员自觉地认同肩负的责任并愿意为此目标共同奉献。在测试人员的工作中，团队不仅是指测试人员之间应该有团队意识，作为项目的一分子，测试人员和开发人员也是一个团队。只有整个项目具备了团队精神，项目才能获得最终的成功。

8．工作热情

测试活动充满了艰辛和挑战。测试人员只有始终保持积极的态度和工作热情，才能够出色地完成各种测试任务和接受各种挑战。没有激情的测试人员，只会满足于完成基本的

测试任务；而充满激情的测试人员不仅能够完成基本测试任务，还能够用积极的态度思考测试过程中遇到的各种问题，努力寻找解决方案，创造性地解决这些问题。同时还可以积极地对自己进行反省，不断地寻找团队和自己的不足，从而进行持续的改进。

7.2.3　个人技能评估

如前面章节所述，测试团队通常由不同的角色组成，例如，测试经理、测试设计人员（测试分析师、技术测试分析师）、测试自动化人员、测试环境管理员和测试执行人员等。测试团队中的不同角色，需要承担不同的职责和具备不同的技能，例如，测试经理需要对整个测试过程中的所有测试活动进行计划和跟踪，因此，测试经理在管理能力、沟通技巧和团队建设等技能方面的要求会更高一些；而测试执行人员主要关注的是测试环境搭建、测试用例执行、测试状态和结果报告等，因此，对他们在测试对象的背景知识和测试的基础知识等方面要求会多一些，而测试管理方面的能力要求要少一些。

个人技能的评估是测试经理的一项重要工作。通过技能的评估能够发现团队成员各自的特性，有利于最大限度地发挥团队的战斗力，同时也可以针对团队成员的不足安排一些练习或者培训，来提高整个团队的能力。对测试团队成员的评估需要从多个角度来进行：软件系统的使用、专业领域和业务知识、系统开发知识、软件测试和软技能等。

通过调查表对测试团队成员进行个人技能评估是个比较有效和直观的方式。下面介绍在 iBAS 项目中，针对测试团队成员进行个人技能评估时所采用的调查表。该项目的测试团队评估从三个维度进行：软技能、测试技能、专业领域和开发知识。每个维度都包括一些更详细的子技能分类，每项子技能对应的评分有 4 级，分别是：0（无）、1（了解）、2（理解）和 3（掌握），它们的含义如下。

（1）0（无）表示被评估人完全不具备该技能；

（2）1（了解）表示被评估人对该技能有一定的认识，并能够在工作中少量应用；

（3）2（理解）表示被评估人基本掌握该技能，并能够在工作中应用；

（4）3（理解）表示被评估人完全掌握该技能，在工作中能够灵活地应用，并不断改进。

评估方式采用测试人员自评和其他相关人评估结合进行。自评是指被评估人按照评估表格中的技能对自己的能力进行自我评估，给出一个结果；相关人评估是指通过熟悉被评估人的相关人对被评估人进行评估，相关人可以是不同的项目利益干系人，例如，被评估人的经理、被评估人的下属或同事，甚至是客户。最后结合所有人的评估结果，给出综合评分。这个评估结果需要和被评估人进行沟通，得到被评估人认可后成为最终的评估结果。整个评估过程需要由独立公正的评估团队来组织和监控，以保证整个评估过程的公正和公平。

1．软技能

软技能主要从创新能力、分析能力、沟通技巧、团队精神和工作热情等几个方面进行评估，软技能评估模板如表 7-3 所示。测试人员的软技能和测试活动并不直接相关，但是它们将影响测试活动的完成效率和有效性。软技能的详细内容参见 7.2.2 节。

表 7-3 软技能评估模板

测试团队成员评估（1）—— 软技能评估				
0（无）：完全不具备该技能 1（了解）：对该技能有一定的认识，并能够在工作中少量应用 2（理解）：基本掌握该技能，并能够在工作中应用 3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进				
技能	0（无）	1（了解）	2（理解）	3（掌握）
创新能力				
分析能力				
沟通技巧				
团队精神				
工作热情				

2．测试技能

测试技能评估主要对测试人员掌握的软件测试专业知识进行评估。测试技能按照测试过程分成 5 个部分：测试计划和控制、测试分析和设计、测试实施和执行、评估出口准则和报告以及测试结束活动。每个部分中又列出了该阶段需要具备的详细的测试技能。通过应用该评估表格，能够比较准确地分析出测试团队成员在测试专业知识方面的强弱项，以及技能的分布情况。测试技能评估表格如表 7-4 所示。

表 7-4 测试技能评估模板

测试团队成员评估（2）—— 测试技能					
0（无）：完全不具备该技能 1（了解）：对该技能有一定的认识，并能够在工作中少量应用 2（理解）：基本掌握该技能，并能够在工作中应用 3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进					
技能		0（无）	1（了解）	2（理解）	3（掌握）
测试计划和控制	制定测试执行进度				
	测试团队建设				
	监控测试过程				
	制定测试计划				
测试分析和设计	系统需求和设计评审				
	测试需求分析				
	编写概要测试用例				
	测试自动化设计				
	编写测试设计规格说明				
	静态测试技术				
	测试设计技术				
测试实施和执行	编写详细测试用例				
	测试环境管理				
	测试环境搭建				
	测试用例执行				
	使用缺陷管理工具				
	使用配置管理工具				
	自动化测试工具使用				
评估测试出口准则和报告	评估测试出口准则				
	编写测试报告				
测试结束活动	测试数据收集分析				
	测试经验教训收集分析				

测试团队中的不同角色要求掌握的技能显然是不一样的。测试经理对于测试计划和控制、评估测试出口准则和测试总结报告以及测试结束活动中的技能最好能够达到级别 3（掌握），而对于测试分析和设计、测试实施和执行中的技能达到 1（了解）或 2（理解）就可以了，甚至有些测试技能对测试经理来说完全不清楚也是有可能的，例如，对于自动化测试设计，测试经理没有这方面的知识，但是团队中相关的技术测试分析师掌握了该技能，那么测试团队也能够很好地完成测试任务。测试设计人员重点要掌握的是测试分析和设计过程中的技能，对于其他过程中的技能要求要低一些。而对于测试环境管理员，重点掌握测试环境管理和搭建相关的技能就可以了，其他的技能只需要基本了解。

3．专业领域和开发知识

作为测试团队的成员，除了要具备软技能和测试专业知识之外，还需要具备专业领域和开发知识。专业领域和开发知识是和被测试对象直接相关的。表 7-5 中的技能是根据 iBAS 项目的实际情况制定的评估表格，里面很多技能都是和 iBAS 产品直接相关的。iBAS 项目中的专业领域和开发知识的评估涉及宽带业务知识、协议相关知识、操作系统和编程语言等 4 类技能，而每个分类中都有多个详细的技能。

单个测试人员完全掌握这些技能显然是不现实的，不同的角色需要重点掌握的技能是不同的。即使同样针对测试设计人员，要求也可能是不同的。因为 iBAS 产品功能很多，每个测试设计人员只需要掌握其中的一个或几个功能，例如，DHCP 模块的测试设计人员对于 DHCP 的协议肯定需要达到级别 3（掌握）；而 IGMP 模块的测试设计人员对于 IGMP 需要达到级别 3（掌握），而对于 DHCP 只要达到级别 1（了解）或级别 2（理解）就可以了。

表 7-5　专业领域和开发知识评估模板

测试团队成员评估（3）—— 专业领域和开发知识

0（无）：完全不具备该技能
1（了解）：对该技能有一定的认识，并能够在工作中少量应用
2（理解）：基本掌握该技能，并能够在工作中应用
3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进

技能		0（无）	1（了解）	2（理解）	3（掌握）
宽带业务知识	网络基础知识				
	宽带城域网结构				
	智能网基础知识				
协议相关知识	TCP/IP 基础知识				
	DHCP				
	IGMP				
	RIP				
	OSPF				
	RADIUS				
	PPP/PPPoE				
	IPv6				
操作系统	Windows				
	Linux				
	UNIX				
	Mac OS				
编程语言	C				
	C++				
	Java				
	UML				
	Perl				
	Shell				
	TCL				

7.3 测试团队动力

选择和招聘员工是测试经理最重要的管理职责之一。除了角色和职责要求的一些特定技能外，还有很多因素要考虑。当选择某人加入测试团队时，必须考虑团队的动态性。测试团队成员技能和性格的多样性，对于提高整个团队的能力非常重要。强大的测试团队能够处理各种不同复杂程度的项目，同时也能成功地处理与其他项目组成员之间的人际关系。

测试过程中测试人员往往压力很大，例如，软件开发进度延期，导致测试进度被不现实地压缩；或对测试团队期望过高，有时甚至是不合理的。测试经理必须招聘能应对高压力环境下工作的测试人员，在面对挫折时能够转移压力，并且在看似不可能的进度安排下也能专心工作。尽管处理进度安排和期望值等问题属于测试经理的职责，但是测试团队成员同样也面临这些压力。因此在为测试团队招聘时，个性能够适应工作环境也是很重要的考虑因素。特别在测试任务繁重，时间紧迫的情况下，在完成自身测试任务后能够主动承担后续任务的人，是测试经理应该寻找的合适人选。

新的成员应该能够得到充分的指导，并快速融入到测试团队。团队中的每个成员都应该赋予特定的角色和相应的职责。角色和职责的定义可以建立在个人评估结果的基础之上，其目的是为了在团队获得成功的同时，个人也能获得成功。为了实现团队和个人双赢的局面，需要将团队成员的个人性格和团队中定义的角色进行匹配，同时在考虑个人天赋的基础上不断提高他们的技能。

需要注意的是：没有人是完美的。一个强大的团队应该建立在不同成员的长处和短处的平衡之上。团队的交叉培训有助于维持和建立团队知识，并增加灵活性。

7.3.1 团队性格角色分类

Meredith Belbin 和他的团队在英国的 Henley 管理学院（Henley Management College）进行了长达 9 年的研究，对来自世界各地的参与者行为进行了分析。参与者接受一系列的心理测试，通过这些测试对参与者的性格特征和行为进行评估。通过研究发现，参与者的行为主要分为 9 类，每一类行为对应一种团队角色（Team Roles）。同时，研究发现，团队中不同团队角色的组合会影响团队的成功。

Belbin 归纳总结的团队角色共有 9 个，它们分别是：智多星 PL（Plant）、外交家 RL（Resource Investigator）、协调者 CO（Co-ordinator）、鞭策者 SH（Shaper）、审议员 ME（Monitor Evaluator）、凝聚者 TW（Teamworker）、执行者 IMP（Implementer）、完美主义者 CF（Completer Finisher）和专家 SP（Specialist）。同时，这 9 个团队角色可以分为三个不同的方向，分别是：行动（Action）、社交(Social)和思考(Thinking)。具体的分类如表 7-6 所示。

表 7-6　团队角色和方向

行　动	社　交	思　考
完美主义者 CF	协调者 CO	审议员 ME
执行者 IMP	外交家 RL	智多星 PL
鞭策者 SH	凝聚者 TW	专家 SP

团队中的这 9 个角色没有好坏之分，只是根据一些行为和性格特征的不同而进行的分类。不同的角色具有不同的特点，有他们各自的强项，也有他们的弱项。表 7-7 是 Belbin 对这 9 个团队角色的特征的归纳总结[①]。

表 7-7　团队角色描述

团 队 角 色	强　项	弱　项
智多星 PL Plant (Innovator/Inventor)	创造性、充满想象力、不拘一格、 解决难题	忽略细节 沟通中过于强势
外交家 RL Resource investigator (Trailblazer/ Switchman)	性格外向、热情、善于沟通、 善于发现机会、开拓人脉	过于乐观 热情缺乏持续性
协调者 CO Co-ordinator (Chairman/Integrator)	成熟、自信、很好的主席、 目标清晰、促进做出决定、好的代理人	可能被人操纵 掺杂个人利益
鞭策者 SH（Shaper）	挑战权威、充满活力、能够承受压力、 勇于克服困难	容易被激怒 给他人带来不愉快
审议员 ME（Monitor-Evaluator）	冷静的、有策略的、有识别能力、发现 所有可能的观点（选择）、准确的判断	缺乏感染力或影响力
凝聚者 TW（Teamworker）	合作的、平和、感觉敏锐、老练、 善于倾听、善于团队建设、消除矛盾	缺乏决断力
执行者 IMP（Implementer）	遵守纪律、行为稳健、可靠、高效、 善于把想法付诸行动	有点儿顽固 对新问题的处理反应迟缓
完美主义者 CF （Completer Finisher）	勤劳、尽职、充满渴望、 能够发现错误和遗漏、按时完成工作	过度焦虑 不愿意成为代理人
专家 SP（Specialist）	诚实、自觉、专注、 技术能力强	贡献的面比较窄 过于专注于技术

表 7-7 中的 9 个性格角色，每个角色都有自己的强项，也都有各自的弱点。性格角色本身没有好坏之分，因此，测试经理在安排测试任务的时候，需要根据不同角色的特点安排不同的工作。那是不是每个团队成员只能对应其中的一种团队角色呢？回答是否定的。虽然每个角色的特征都不一样，但是在现实生活中，每个人都可能对应其中的多个角色，例如，某人可能同时具备“凝聚者 TW”和“协调者 CO”的所有特点，同时还具有“专家 SP”这个团队角色的优点。而且团队成员在从事不同工作的时候，其表现出来的特点也不同。当一个具有“凝聚者 TW”和“协调者 CO”所有的特点，同时还具有“专家 SH”优点的人从事测试设计工作的时候，他表现出来的更多的是“专家 SH”的特点，而当他成为一个测试团队的负责人后，可能表现出更多的是“协调者 CO”和“凝聚者 TW”的特征。

① 关于 Belbin 研究的团队角色分类，请参考 *Belbin_Team_Role_Descriptions*（www.belbin.com）。

成功的团队必须由多种不同的团队角色组成，任何单一的团队角色都不利于团队的成功。所以测试经理在组建测试团队或者招募新员工的时候，要考虑测试团队中团队角色的分布情况，避免团队角色分布过于单一。

7.3.2 案例：测试团队分析

针对测试团队进行客观的技能评估，可以帮助明确当前团队的强项和弱项，从而可以有针对性地决定招聘什么类型的测试人员。评估的方式可以是面试、考核候选人、评审候选人工作产品等手段。而需要评估的技能包括硬技能和软技能。

硬技能（或者技术技能）可以通过下列活动体现。

（1）评审技术文档（需求规格说明、代码等），并以清晰易于理解的方式客观地提出评审意见；

（2）根据需求规格说明生成测试用例，或者根据给定的场景从多种测试技术中选择适用的测试技术生成测试用例（例如，使用决策表来测试一系列业务规则）；

（3）评估失效并准确地记录；

（4）阐述对缺陷分类信息的理解；

（5）阐述对缺陷根本原因的理解；

（6）使用工具测试给定的 API，包括正向测试和逆向测试；

（7）使用 SQL 发现并修改数据库信息以测试一个给定的场景；

（8）设计可以执行多种测试并收集测试结果的测试自动化工具；

（9）执行自动化测试并排除失效；

（10）编写测试计划和规格说明；

（11）编写包含测试结果评估的测试总结报告。

软技能（人际交往能力）可以通过下列活动体现。

（1）为利益干系人演示落后于进度的测试项目的信息；

（2）向坚信无缺陷的开发人员解释提交的缺陷报告；

（3）为同事培训被测软件产品相关的业务知识；

（4）向管理层演示一个由于无效过程而导致的问题；

（5）评审同事编写的测试用例，并向其沟通自己的意见；

（6）面试测试招聘候选人；

（7）表扬开发人员。

上述的硬技能和软技能是比较常见的技能要求，并不是完整的列表，测试团队所需的技能会随着环境和组织的变化而变化。测试经理通过建立有效的面试问题，结合技能演示，可以评估测试人员的技能，并确定他们的强弱项。建立的良好评估机制，也可以应用于所有的团队成员，帮助确定发展和培训的领域。

结合上面的硬技能和软技能列表，下面以 iBAS 为例，讲解如何对测试团队进行分析，并根据分析结果招募有特定技能的新员工。

iBAS R1.0 项目测试经理需要新招募一名测试分析师。为了能够招募到合适的测试分析师，他首先对团队中现有的测试分析师进行了分析，分析使用 7.2.3 节中提及的评估表格，从软技能、测试技能、专业领域和开发知识这三个方面进行分析。iBAS R1.0 项目测

试团队目前有测试分析师三名。他们的评估结果如表 7-8～表 7-10 所示。

表 7-8　软技能评估实例

测试团队成员评估（1）——软技能评估				
0（无）：完全不具备该技能 1（了解）：对该技能有一定的认识，并能够在工作中少量应用 2（理解）：基本掌握该技能，并能够在工作中应用 3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进				
技能	测试分析师 A	测试分析师 B	测试分析师 C	总分
创新能力	2	3	2	7
分析能力	3	3	3	9
沟通技巧	3	2	1	6
团队精神	2	2	2	6
工作热情	2	2	2	6

表 7-9　测试技能评估实例

测试团队成员评估（2）——测试技能					
0（无）：完全不具备该技能 1（了解）：对该技能有一定的认识，并能够在工作中少量应用 2（理解）：基本掌握该技能，并能够在工作中应用 3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进					
技能		测试分析师 A	测试分析师 B	测试分析师 C	总分
测试计划和控制	制定测试执行进度	1	0	0	1
	测试团队建设	2	1	1	4
	监控测试过程	2	1	1	4
	制定测试计划	1	0	0	1
测试分析和设计	系统需求和设计评审	3	2	2	7
	测试需求分析	3	3	3	9
	编写概要测试用例	2	3	3	8
	测试自动化设计	1	0	0	1
	编写测试设计规格说明	2	2	2	6
	静态测试技术	3	2	2	7
	测试设计技术	2	3	3	8
测试实施和执行	编写详细测试用例	2	2	2	6
	测试环境管理	1	1	1	3
	测试环境搭建	1	1	1	3
	测试用例执行	2	2	2	6
	使用缺陷管理工具	2	2	2	6
	使用配置管理工具	2	2	2	6
	自动化测试工具使用	0	0	0	0
评估测试出口准则和报告	评估测试出口	1	0	0	1
	编写测试报告	1	0	0	1
测试结束活动	测试数据收集分析	1	1	1	3
	测试经验教训收集分析	1	1	1	3

表 7-10　专业领域和开发知识评估实例

测试团队成员评估（3）——专业领域和开发知识					
0（无）：完全不具备该技能 1（了解）：对该技能有一定的认识，并能够在工作中少量应用 2（理解）：基本掌握该技能，并能够在工作中应用 3（掌握）：完全掌握该技能，在工作中能够灵活应用并不断改进					
技能		测试分析师 A	测试分析师 B	测试分析师 C	总分
宽带业务知识	网络基础知识	3	2	2	7
	宽带城域网结构	3	2	2	7
	智能网知识	1	3	2	6
协议相关知识	TCP/IP	2	2	2	6
	DHCP	3	1	1	5
	IGMP	3	1	1	5
	RIP	1	0	0	1
	OSPF	1	0	0	1
	RADIUS	1	3	2	6
	PPP/PPPoE	3	1	3	7
	IPv6	1	1	1	3
操作系统	Windows	3	2	2	7
	Linux	1	3	2	6
	UNIX	1	3	2	6
	Mac OS	0	0	2	2
编程语言	C	1	1	1	3
	C++	1	0	1	2
	Java	1	0	1	2
	UML	1	0	0	1
	Perl	0	0	0	0
	Shell	0	1	1	2
	TCL	0	0	0	0

从分析结果看，该测试团队的测试分析师的技能水平较高，能够胜任大部分测试分析相关的任务。同时，测试分析师的软技能也都很高，测试技能中与测试分析师需要关注的测试分析和设计以及测试实现和执行的大部分技能，该团队表现也很好。专业领域和开发知识中宽带接入领域相关技能掌握很好，而协议和操作系统部分需要提高，目前由于产品只实现了部分功能，现在掌握的技能还能够满足需要，但是随着产品后续 IPv6 以及路由协议等功能的加入，这三名测试分析师将会遇到较大的挑战；编程语言方面整体都比较弱，有待提高。由于测试分析师介入开发活动还不是很深，以及目前测试团队自身的测试工具开发需求较少，因此，编程方面的改进需求目前并不强烈。

目前，测试分析师的整体情况如上所述。在现有的三名分析员的基础上，由于 iBAS 系统中功能的不断增加，计划招募一名新的分析员。考虑后续新增加的功能包括 IPv6 和路由协议，以及整个测试团队对自动化需求越来越大，因此，现招募的测试分析师最好能够符合如下需求。

（1）具备良好的软技能；

（2）具备基本的测试分析和设计相关技能；
（3）具备基本的测试实现和执行相关技能；
（4）掌握测试自动化相关技能；
（5）具备良好的宽带接入基础知识；
（6）熟悉 IPv6 和各种路由协议；
（7）熟悉 UNIX 和 Mac OS 的操作；
（8）至少掌握一种脚本语言（Perl、Shell 或 TCL 等）。

同时，测试经理发现现有的三名测试分析师虽然各项技能掌握很好，但是性格都偏内向，团队气氛不够活跃，所以希望条件允许的话优先招募一名性格较外向的测试分析师。

如果完全按照上面的招募要求，测试经理可能很难招到合适的测试分析师。同时具备上面提及的所有技能是比较困难的，在实际的招募过程中必须进行取舍，挑选最符合上面要求的人员。有一些技能如果暂时不具备，招募进来之后可以通过培训或者自学等方式不断增强。

这里对测试团队成员的分析中，虽然每个成员的相关技能都用数字 1、2、3 来表示，但是本质上是定性的分析，并没有对测试人员进行定量的分析。关于定量评估的内容，参见 7.5.2 节。

7.3.3　测试团队优化

测试经理一方面需要有效地管理测试团队，另一方面，又需要处理测试过程中面临的各种不同问题和挑战。对测试团队的发展产生负面影响的行为，不仅会对测试活动产生严重的影响，也可能会对测试质量和测试进度等产生不利影响。常见的测试团队管理挑战如下。

（1）专业技能：测试团队成员缺少软件产品相关的技能和知识。
（2）过程：测试团队成员对测试过程和开发过程等不熟悉。
（3）角色和职责：测试团队成员分工不明确，人员责任不清楚。
（4）对外合作：测试团队成员和其他团队之间合作困难。

当测试团队面临这些问题的时候，应该如何来解决这些问题？下面针对这些问题提供一些基本的思路、方法和建议。

1. 专业技能

软件测试活动的开展需要测试人员具备多方面的知识和技能，例如，产品相关的背景知识、软件产品采用的技术和方法标准、测试需要使用的工具（例如缺陷管理工具）等。掌握这些相关知识是测试人员开展测试活动的基础，假如这方面的知识和技能欠缺，那么进行测试活动将非常困难，甚至是不可能的事情，从而降低测试效率和测试质量。

每个测试人员具备的经验各不相同，通过彼此共享交流，可以使得整个测试团队的相关技能水平得到提高。在实际工作中，测试人员应当不断总结经验，并与其他测试人员交流，相互取长补短，从而不断提高专业技能。以下是可以采用的几种共享知识经验的方式。

（1）团队内技术交流和讨论；
（2）让经验丰富或技术好的团队成员进行指导或举办讲座；

（3）建立测试部门的小图书室或图书俱乐部，测试团队通过阅读测试方面的书籍，可以在就餐时间或其他时间进行讨论和交流。

上面提到几种共享知识经验的方式可以称为非正式的培训活动。也可以采用正式的培训活动提高测试团队的专业技能，例如，参加外部的专业培训班、正式的测试技术研讨会、专家提供的培训课程，以及公司内部组织的各种培训等。

2．过程

测试人员除了需要了解和掌握产品相关的技术和技能知识以外，还需要了解组织和项目层面的具体过程，包括组织质量方针、开发模型、测试过程、缺陷管理过程以及配置管理过程等。

测试团队成员主要通过参加培训的方式了解相关过程，例如，测试人员通过培训，可以了解组织内测试过程的不同阶段、测试的主要活动、测试的主要输入和输出，以及和其他团队之间的相互关系等。测试人员也可以参加一些测试认证（如 ISTQB 提供的认证培训），通过测试认证培训，测试人员可以对整个测试知识体系有一个全面和深入的了解。

3．角色和职责

角色和职责不明确，会导致测试工作的混乱无序，同时也会挫伤测试人员的工作积极性。角色和职责包含两个方面的内容，首先是定义明确的测试团队的组织架构，其次是定义完整的测试角色和相关的职责。角色和职责的详细内容，参见 7.2.1 节。

4．对外合作

对外合作包括测试团队和开发团队、系统人员、维护人员，以及客户等之间的交流和沟通。

对外合作的重点是团队之间的沟通和理解。每个人对技术和知识的理解不同，导致对工作产品的认识也各不相同。另外，每个团队在项目中担当的角色不同，项目中担当不同角色的团队在沟通时也会有障碍，例如，对于构建软件产品而言，开发人员是“创造性”的活动，而测试人员在很多时候是“破坏性”的活动，如何在这两者之间进行平衡，是测试经理需要关注的焦点。

对外合作的另一个关注点是项目的利益干系人应该在共同的平台上进行交流，例如，单个项目应该尽量采用统一的项目开发过程、缺陷跟踪过程、需求变更过程等。只有大家在这些过程上达成一致，才能在面临这些问题的时候进行更好的沟通。



7.4 使测试适合组织

优秀的测试团队是做好软件测试工作的基础。测试活动的开展必须贯穿于整个软件开发生命周期，因此，测试活动应该和相关的开发活动进行协调，并统一规划。不同的组织，会以不同的方式组织测试团队，以达到组织和项目层面的测试目标。

采用独立的测试团队进行测试和评审，发现缺陷的效率会明显提高，因此，独立的测试团队是一个较好的选择。独立测试的方式并不是完全的替代开发人员进行的测试，因为开发人员也可以高效地在他们的工作产品中找出很多缺陷。根据组织和项目的特点，可以选择不同的独立测试策略。

（1）非独立的测试人员（开发人员测试自己的代码）；

（2）测试由不同的开发人员执行，而不是代码的编写者本人；

（3）测试由开发团队中专门的测试人员完成；

（4）测试由独立于开发团队的测试团队完成；

（5）外部测试专家基于特定的测试目标执行测试；

（6）测试由组织外的团队执行（例如外包）。

对于庞大、复杂的系统或安全关键系统，最好让独立的测试人员进行多个级别的测试。开发人员也可以参与其中（特别是在测试级别比较低的时候），但是开发人员缺少测试目的性和测试相关技能会限制他们的测试效率和有效性。独立测试人员有权要求定义测试过程及规则，但是只有在明确授权的情况下才能充当这种过程相关的角色。在实际的项目中，可以考虑采用多种独立测试策略相结合的形式。

1．非独立的测试人员

在这种情况下，没有任何的独立性。开发人员对自己开发的代码进行测试。开发人员将根据自己的理解对代码进行测试，一旦发现缺陷，开发人员能够马上进行修复，大大缩短解决问题的时间；同时由于开发人员对代码很熟悉，知道哪些地方可能存在更多的问题，可以有针对性地进行测试。但通常情况下，开发人员愿意花费更多的时间在开发活动上，而留给测试活动的时间很少，测试活动的质量无法得到保障。从心理学来看，让开发人员发现自己代码中存在的缺陷存在一定难度，因此，不太容易发现缺陷。

2．测试由不同的开发人员执行

测试由不同的开发人员执行，而不是代码的开发者本人。这种形式在一定程度上体现了测试的独立性。一方面，担当测试工作的开发人员对系统设计和开发的代码比较熟悉，而且同为开发人员，沟通比较通畅；另一方面测试对象并不是自己开发的代码，这种情况下缺陷的发现率将会有一定程度的提高。开发人员兼任测试人员的工作，没有形成独立的测试团队，整个团队的核心是开发，这种情况下，一个开发人员测试另一个开发人员的代码可能不情愿报告缺陷。同时，开发人员设计的测试用例通常集中在正向的功能测试用例上，对于一些非功能测试以及异常情况的考虑比较少。

3．测试由开发团队中专门的测试人员完成

开发团队内部有专门的测试人员或测试团队，这些测试人员或测试团队向开发经理汇报工作。该模式下，测试人员已从开发人员中独立出来，因此，测试人员具有一定的独立性，他们可以采用和开发人员不同的视角分析和检查被测试产品；同时测试人员和开发人员联系紧密，可以和开发人员及时沟通。但是这种模式下，测试人员仍然受到开发经理的制约，不能完全独立地从产品质量出发进行测试，由于整个项目进度和经费的关系，测试的投入可能不够，在开发任务紧张的情况下，这些测试人员还可能负责部分开发任务，从而不能保证产品质量。

4．测试由独立于开发团队的测试团队完成

这种情况下，测试团队具有相当的独立性。此时，测试团队直接向利益干系人（例如项目经理）汇报，测试团队重点关注被测试对象的质量。发现缺陷成为测试团队最重要的目标之一，对测试对象中出现的问题能够进行客观的分析和评价。由于整个团队都负责测试相关活动，测试团队可以集中精力培养整个团队的测试能力，成为专业的测试团队。

由于测试团队独立于开发团队，测试团队成员的开发技能相对较弱，对于系统的需求、设计和代码的理解需要投入更多的工作量，而且测试人员和开发人员需要建立正式的沟通渠道，例如，需要建立更加完善的配置管理和缺陷管理系统。

5．外部测试专家基于特定的测试目标执行测试

聘请来自于组织外部的专家对特定的测试目标执行测试。外部专家并不对整个系统进行全面的测试，而只关注其中特定的领域，例如，可能只关注被测试对象的重要的质量特性（例如：易用性、安全性或性能等）。这些专家在特定的领域都有很深的造诣，可以在短时间内快速对被测试对象的特定方面进行评估，发现缺陷并提出改进意见。但是这些外部的测试专家的成本通常都很高，项目因此要有高额的投入。

6．测试由组织外的团队执行

这种形式实现了最大化的测试独立，既能够以揭露软件中的缺陷为目的开展工作，也能不受发现的缺陷的影响。组织和经济上的独立性使其有更充分的条件按测试要求去完成工作。由于专业优势，独立测试工作形成的测试结果更具信服力。由于测试结果常常和软件的质量评价联系在一起，因此，由专业化的独立测试机构进行评价权威性更强。但是这种独立性也带来一些负面影响：知识传递可能不充分、需要明确的需求定义、需要很好的沟通平台以及外部组织的质量不确定等。这些影响因素在采用这种形式的测试组织时是需要加以考虑的。

上面介绍了 6 种不同的测试独立性的形式。测试独立性的优点如下。

（1）对软件测试和软件中的错误抱着客观的态度，这种客观的态度可以解决测试中的心理学问题，既能够以揭露软件中的缺陷为目的进行工作，也能不受发现的缺陷的影响。

（2）经济上的独立性使其工作有更充分的条件按测试要求去完成。

（3）独立测试作为一种专业工作，在长期的工作过程中势必能够积累大量的实践经验，形成自己的专业优势。

（4）软件测试是技术含量很高的工作，需要有专业队伍加以研究，并进行工程实践。专业化分工是提高测试水平、保证测试质量、充分发挥测试效用的必然途径。

（5）由于专业优势，独立测试工作形成的测试结果更具信服力，而测试结果常常和对软件的质量评价联系在一起。因此，由专业化的独立测试机构进行评价的权威性更强。

（6）独立测试机构的主要任务是进行独立测试工作，这使得测试工作在经费、人力和时间方面更有保证，不会因为开发的压力而减少对测试的投入，降低测试的有效性，从而可以避免开发组织侧重软件开发而对测试工作产生不利的影响。

测试的独立性并不是越高越好。随着测试独立性的提高，也会带来以下一些问题。

（1）整个组织的复杂度越来越高，管理成本增加，例如，当测试团队属外包测试时，对于测试团队的测试质量无法及时监控。

（2）沟通效率降低，原来只是需要打个招呼的问题，现在需要通过复杂的配置管理、缺陷管理和文档管理系统来解决。

（3）测试人员和开发人员的距离越来越远，项目团队之间的工作氛围会下降，某些极端情况下甚至出现开发人员和测试人员的对立现象。

（4）测试人员重点关注测试相关技能，对开发技能的掌握比较差，不利于发现系统需求和设计方面的缺陷。

（5）独立的测试团队可能降低开发人员的质量责任感，开发人员可能会觉得产品质量应该是测试团队的事情，而不是整个项目团队的责任。

一个组织的测试活动可以采用多种测试独立性策略。不同的测试级别可以使用不同形式的测试独立性策略。低级别的测试采用独立性比较低的策略，高级别的测试采用独立性比较高的策略，例如，组件测试和集成测试由开发人员来完成，系统测试由企业内部独立的测试团队完成，验收测试由组织外引入的测试团队完成，如表 7-11 所示。具体的形式可以根据组织和项目的实际情况进行调整。

表 7-11　测试团队和测试级别

级别	独立性		
	不同的开发人员	组织内独立的测试团队	组织外部的测试团队
组件测试	×		
集成测试	×		
系统测试		×	
验收测试			×

7.5 激励

7.5.1 激励方式

在整个测试过程中，人是最重要的因素之一。不管是多好的过程，还是多好的方法，最终都需要由人来执行和完成。好的过程就像是一条高速公路，而人就是在这条高速公路上的车，好车配合好路才能够保证行驶的速度和安全。

测试经理在日常工作中的一个重要任务就是保持对测试团队成员的激励。一谈到激励，可能很多人下意识就会想到金钱，例如，给员工奖金，或者加工资。实际上，物质奖励只是激励方式的一种，但这并不是唯一的和最有效的方式。

每个员工都有自己的梦想和目标，希望能够获得更多自主的空间，希望自己的才能得到施展，希望得到认可，希望自己的工作富有激情，因此，测试经理除了提供物质上的激励以外，更应该为员工提供一个施展他们才能的舞台，激发员工内在的自我激励和自我实现。而且每个员工的需求不一样，激励的方式也各不相同。例如，公开表扬对有的员工有明显的激励作用，但对于性格内向的员工，效果可能会大大打折；提供弹性的工作时间也是对员工的一种激励①。下面介绍一些常见的激励方式。

1．满足员工的需求

作为团队核心的测试经理，必须针对部门内员工的不同特点“投其所好”，寻求能够激励他们的动力。每个人内心需要被激励的动机各不相同，因此，奖励杰出工作表现的方法也应因人而异。不同的人在不同时期的需求也是不一样的。员工主要的需求包括：

---

① Larry R Williams 在 *Keep'em Motivated: A Practical Guide to Motivating Employees*（Marshall Cavendish，2003）一书中对如何激励员工进行了详细的描述，读者可以从中获得更详细的信息。

（1）得到尊重和认可。
（2）获得自我表现的机会。
（3）在积极的团队氛围中工作。
（4）获得一定的做决定的权力。
（5）公平的竞争环境。
（6）广阔的成长空间等。

要满足员工的需求，首先需要理解员工。这就要求测试经理在日常的工作和生活中能够多倾听、多观察，还应该注意员工的一些肢体语言或者异常的情绪波动等。测试经理可以通过以下途径获得员工的具体需求：积极倾听员工的呼声、换位思考、观察员工的情绪波动并做出积极的回应等。当员工的需求获得满足时，他们的工作效率会有很大的提高，员工的内在激励会起到重要的作用，团队会处于一个积极向上的氛围，员工的聪明才智和创造性才能够得到最大的发挥。

马斯洛需求层次理论，也称“基本需求层次理论”，由美国心理学家亚伯拉罕·马斯洛提出[①]。马斯洛是当代最伟大的心理学家之一，由于他的著作使人本心理学的观点更加丰富和清晰，所以被人们称为“人本心理学之父”。他的需求层次理论对团队激励有着很大的帮助。表 7-12 列出了根据马斯洛的需求层次理论得到的团队激励的实例。

表 7-12 马斯洛的需求层次理论在团队激励中的实例

马斯洛需求层次	工作中的对应关系
自我实现的需要	参与挑战性的项目、创造性获得鼓励和支持、获得创新的机会
尊重的需要	获得尊重和认可、能够被委以重任
情感和归属的需要	友好的团队氛围、良好的客户关系
安全上的需要	工作的安全性、良好的工作环境
生理上的需要	基本的薪酬、工作空间、饮食

2. 有效授权

测试经理在安排工作的时候，应该考虑让团队成员承担更多的责任并拥有更多的权力。授权并不一定是升职。测试经理在向其测试团队成员分配工作时，也要授予他们相应的权力，否则就不算授权。测试经理应该在合适的场合让所有的相关人员知道被授权者的权责，同时需要确保授权之后应该尽量减少干涉，要给被授权员工做相应决策的权力。有效授权是一种很好的激励员工的方法。权力的下放同时也意味着责任的下放。员工在获得一定的决策权力的同时，也应该明白要全力以赴把事情做好，要对任务的结果负责。通过有效授权，员工的积极性和创造性能够被最大限度地调动起来，同时员工的责任感和主人翁精神会得到加强，对组织的认同感会不断提高。员工通过承担更多的责任，也获得提高自身各项技能的机会，从而获得更广阔的发展空间。但是在授权的过程中，测试经理也要注意确保授权的范围和员工的实际能力相匹配，尽量避免一开始就赋予团队成员过多的授权和责任。在实际的授权过程中，测试经理可以以渐进的方式授予测试团队成员的权力，同时在必要的时候需要为员工提供相应的培训。

☆示例：有效授权

测试经理通常需要协调整个测试过程中的所有活动，包括测试计划和控制、测试分析和设计、测试实现和执行、评估测试出口准则和测试总结报告以及测试结束活动

① A H Maslow. 动机与人格. 许金声，等译. 北京：华夏出版社，1987.

等。但是作为测试经理，并不需要对所有的测试活动亲力亲为，而是应该将测试活动进行划分，将测试过程中的一些活动下放到测试团队成员中去。在项目中可以采用“一事经理”的方式。所谓的“一事经理”，就是对某个独立的任务或活动具有决策权力的人。测试经理可以将测试活动中的测试环境管理、测试用例配置管理以及测试度量等任务的权力授予相应的测试人员，由其负责处理相关事宜，定期向测试经理汇报。

以测试度量为例，某个测试团队成员成为该任务的“一事经理”后，全权负责测试度量相关事宜，包括测试度量系统的设计、测试度量数据收集和测试度量数据分析等，而测试经理作为测试度量活动的一名成员而不是管理者参与到测试度量活动中。测试经理最终使用测试度量活动的分析结果作为其他活动的输入，例如，作为测试出口准则评估的输入。测试人员在“一事经理”这个方式下也得到了锻炼，自己的技能得到了增强，同时获得了整个团队的尊重和认可；测试经理也因此减少了日常的工作量，可以将更多的精力投入到其他测试管理活动中去。

3．有效沟通

沟通无时无地不在，测试经理每天要和各种不同角色的人以各种不同的方式进行沟通。有效的沟通能够节约测试经理的时间和工作量，同时也能帮助团队成员更好地完成任务。另外，有效的沟通还有助于构建一个开放的测试团队，最大限度地发挥团队的积极性。倾听是有效沟通的一个重要方面，尤其是对待团队成员的抱怨，测试经理更应该认真倾听，如果能做到认真倾听，大部分的抱怨都可以得到解决。很多时候，员工可能并没有期望问题得到解决，他们更看重的是测试经理或管理者的态度，通过认真的倾听，员工的问题即使没有得到解决，员工的不满意度也会得到很大的降低。此外，在沟通的过程中，测试经理要避免过于封闭而听不进员工的建议，或者只是进行有选择地倾听，例如，只听好的内容，对提及的问题视而不见，同时，测试经理必须对员工的意见及时进行反馈。

沟通的方式有很多，例如，面对面交流、E-mail、电话、文档评审等。测试经理应该根据实际需要选择合适的交流方式，例如，简短的信息传递，可以通过电话的方式很快解决；对于复杂方案的讨论，需要安排专门的会议；有些事情可以先通过 E-mail 的方式传递基础信息，给员工一个准备的时间，然后通过面谈的方式做最后的沟通。以下的建议将有利于改进沟通的效果。

（1）多听少说。

（2）使用简单、容易理解的语句。

（3）注意使用合适的肢体语言。

（4）不明白的要及时提出问题。

（5）要经常沟通以及沟通要保持热情。

☆示例：有效沟通

测试经理和测试团队成员之间如果能够建立有效的沟通渠道，团队成员和测试经理之间能够有效地交换意见，将有助于测试团队成员的激励。但是在实际的测试过程中，测试团队成员很少主动找测试经理反馈意见。尤其是以技术为主的测试团队中，测试团队成员相对比较“内向”，交流不够主动。这种情况下，测试经理可以定制一个团队沟通交流计划，团队成员每个月都有一次和测试经理单独面谈的机会。测试经

理会轮流和每个测试团队成员进行面对面的交流，给员工一个反馈意见的机会。这些面谈都是在一对一的情况下进行的，团队成员不会有太多的顾虑；同时面谈的时间都是提前计划好的，如果有需要反馈的问题或建议，员工在沟通之前可以做好充分的准备。测试经理在这种方式下要把面谈的气氛调节为一种非正式的交流，可以交谈任何内容，从工作到生活。

4．提供学习和培训的机会

现代社会，科学技术迅猛发展，信息和知识急剧增加，知识更新周期缩短，创新频率加快，各种技术和方法层出不穷。如果员工不积极学习，就很难跟上时代发展的步伐，甚至会被社会淘汰。因此，每个员工自身都有危机感和紧迫感，从而自身都有学习和培训方面的需求和需要。这种情况下，为员工提供良好的学习和培训的机会，显然能够激励员工创造更大的价值。通过提供学习和培训的机会，不仅员工自身的能力得到增强，对组织的认同感提高，整个团队的能力也会得到提升，从而能够更加有效地完成相关任务，并承担更大的责任。学习和培训的方式有很多，可以在团队内部组织交流和学习、相互进行培训，也可以聘请外部专家进行培训；可以通过 E-learning 的方式，也可以通过课堂教学的方式。要开展学习和培训，必须将员工的兴趣和工作需要结合起来。在开展培训之前，首先收集员工的培训需求，然后结合项目或者团队的目标，从而在个人目标和团队目标之间实现共赢。学习或培训后，需要及时收集员工的反馈，不断改进培训的效率和有效性。

☆示例：提供学习和培训的机会

“岗位轮换”是一种很好的学习方式。当测试团队成员在自己的岗位上工作了一段时间以后，相关技能已经熟练掌握，这个时候都希望能够有机会再增加一些其他方面的技能。这种情况下，可以在测试团队内部进行岗位轮换，给每个员工一个学习新知识的机会。在 iBAS R1.0 项目中，随着测试执行的不断进行，被测试对象中发现的问题越来越少，测试人员已经开始“审美疲劳”，觉得很难再发现新的问题。这个时候，在测试团队中进行岗位轮换，原来执行 IGMP 系统测试的人员开始测试 DHCP 功能，其他人也相应地进行轮换。通过这种方式，测试人员可以掌握原来不具备的知识，增加了对整个被测试系统的了解；同时由于相互轮换，起到了一个评审的作用，测试人员对新分配任务的功能可从自己的角度重新思考，可以对原来的测试用例进行修改或补充，这样也有利于发现更多的缺陷，提高被测试对象的质量。

5．尊重和认可

每个人都希望得到别人或团队的尊重和认可。尊重和认可能够使员工对自己更加自信、对工作更加热爱，能够鼓励员工提高工作效率。给员工的认可也要及时而有效，当员工工作表现很出色时，测试经理应该立即给予称赞，让员工感受到自己受到上司的赞赏和认可。测试经理可以通过各种不同的方式认可员工的工作表现，例如，口头赞赏、书面赞美、对员工一对一的赞赏、公开的表扬等，从而不断鼓舞员工士气。对于测试人员而言，当测试人员的建议被开发人员或项目团队采用、测试人员能够在项目早期介入项目开发活动、为测试团队分配足够的资源等，这些都是对测试人员的尊重和认可，可以激励测试人员更好地工作。

☆示例：尊重和认可

“每周之星”是一种表扬员工的方式，通过这种方式可以使员工感受到被组织的认可，从而获得满足感和成就感。测试经理每周发掘团队中表现出色的员工进行表扬，是尊重和认可的重要表现形式。员工的出色表现可以是各种各样的，例如，需求评审过程中发现了很多问题、测试设计过程中有效地引入了测试自动化、为客户制定了良好的解决方案等。测试团队每周肯定都会有一些特别的事情发生，因此，总归可以发掘成为“每周之星”的员工。即使是测试团队处于项目的前期学习阶段，也有很多员工的行为是值得表扬的，例如，对及时总结或者分享学习成果的员工进行表扬。“每周之星”这样的形式不仅对员工的工作进行了认可和表扬，而且能够激励其他员工不断地努力。

6．物质奖励

上面介绍的几种激励方式更多地偏重于精神上的鼓励。实际上，除了精神激励外，物质奖励也是必不可少的，它也是日常工作中经常使用的一种方式。薪水不仅能保证员工生存，同时能者多得的薪酬机制也能有效地起到激励效果。物质奖励的多少依赖于员工能为公司带来的价值，例如，对于为公司创造出高利润、开发出赢利新项目的核心人才，通过加薪激励是必不可少的。在使用物质奖励方式的时候，一个重要的考虑因素是公平。如果某个员工的贡献突出，那么可以对这个员工进行单独的奖励。假如突出的是团队共同努力的结果，那么需要对整个团队进行奖励。同时物质奖励要有充足的理由，确保只有表现优秀的员工或者团队才能够获得物质奖励。

☆示例：物质奖励

物质奖励的方式既可以是单次奖金的形式，也可以是加薪的方式。在平时的激励过程中，测试经理应该综合应用这两种形式。每个季度可以对个人或团队进行一次评比，对表现突出的优秀个人或者优秀团队进行单次的奖金激励。每年对团队成员进行综合考核，根据团队成员的年度表现，对不同贡献的员工进行不同程度的物质奖励，可以是年度的加薪，也可以是年度加薪和单次的年终奖相结合的方式。

上面介绍了几种常见的激励方式，其实，激励的方式还有很多，例如，建立良好的团队氛围、鼓励大家注意身体健康、重视工作和生活的平衡等。在日常管理过程中，测试经理应该灵活应用各种激励方式。

激励测试人员的方式有很多，同时也有很多方式会打击测试人员的积极性，造成消极的影响，例如，测试人员在一个几乎看不到结束期限的项目上工作；测试人员致力于保证产品的质量，并且投入了额外的时间和精力，但是由于一些外部因素影响，输出的产品仍旧没有满足计划中设定的目标；尽管测试人员尽了最大的努力，但还是没有及时完成测试任务；如果测试人员的贡献没有被理解和衡量，不管最终结果是否成功，对测试人员而言都很可能造成消极的影响。

7.5.2　量化管理

测试团队要想在组织内获得尊重和认可，必须能够为组织创造价值。测试团队通过进

行有效的测试活动，能够保证测试对象的质量，有效降低产品的质量风险。测试团队尽可能多地在产品发布前发现缺陷，通过修改缺陷能够提高产品质量，减少遗留缺陷，降低维护成本。如果能够在产品开发的前期发现和修复缺陷，那么降低成本的效果就更为明显。但是测试团队的贡献不能简单地从定性的角度来说明，通过定量的形式明确测试团队在组织内的贡献是测试经理的重要职责之一，从而确保测试团队在组织内获得尊重和认可。只有当测试团队获得尊重和认可，测试团队的成员才可能获得更多的认可。测试经理需要从不同的角度对测试团队的贡献进行量化，例如，从提高产品质量、减少产品成本和降低风险等方面提供具体的数据支持。质量成本分析是用于度量测试的定量价值和效率的成熟方法，详细内容参见 2.7 节。

测试经理在保证测试团队在组织内得到尊重和认可的同时，也需要对测试团队内部进行量化管理。每个测试团队成员都希望获得更多的奖励，这就要求测试经理能够对团队成员进行量化的绩效考核。内部的量化管理需要创建一个公开公平的环境。测试经理和团队成员共同制定考核指标，得到大家的认可后统一执行，并根据每年组织目标的变更进行更新。考核指标有多种多样，常见的指标有设计的测试用例数目、开发的测试脚本代码行、被评审的文档的页数、发现的缺陷数目、发表的文章数目、给其他团队成员培训的次数、遗留的缺陷等。考核指标应该进行量化，例如，每个指标规定一个权重，最后统一考核，测试经理根据考核结果进行相应的激励。量化管理有利于团队成员之间的合作和良性竞争，同时也可以通过制定不同的考核指标来引导团队成员的发展方向，把团队成员的个人目标和组织目标统一起来。

软件测试属于软件研发的一部分，对研发人员的量化考核是一个普遍存在的难题。研发人员的考核不像市场人员那么直接，可以直接用销售额或增长率来考核。表 7-13 是某项目中对测试人员的一个考核表。

表 7-13 测试人员量化考核表

考 核 科 目	考 核 内 容	考 核 结 果
团队建设	提供的培训的时间（小时）	
	编写的总结的数量（篇）	
	获得专利数量（个）	
	获得的认证的数量（个）	
	在公司内部发表的文章（篇）	
	在外部期刊发表的文章（篇）	
测试活动	编写的测试计划（页）	
	编写的测试设计（页）	
	编写的测试规程（页）	
	编写的测试用例（个）	
	发现的缺陷总体权重	
	执行的测试用例（个）	
	自动化测试脚本（代码行）	
用户反馈	开发人员投诉次数（次）	
	遗留到客户的缺陷（个）	
	获得的内部奖励（次）	

该考核表主要关注测试的输出，本着以结果为导向的原则，例如，编写的测试用例的数量、发现的缺陷和提供的培训的时间等。考核的科目分为三个部分：团队建设、测试活动和用户反馈。每个科目的权重各不相同。权重可以随着项目或产品的不同进行动态调整。对测试人员进行量化考核的难点在于，并不是所有的测试活动都能进行量化的，尤其是当考核是以结果为导向的时候。同时，不同的测试人员分工不同，不同的活动之间进行横向比较也存在一定的困难，例如，两个测试人员分别从事自动化测试脚本开发和测试计划的编写，他们之间的输出就很难比较。在量化考核的过程中，要注意考核的目的是为了激发测试人员为客户和公司创造更多的价值，并不是仅仅为了给每个员工进行打分。所以，“绩效管理”在很多时候要比“量化考核”更有意义。

7.6 沟通

测试人员的沟通主要体现在以下三个层次。

（1）测试产品的文档化：测试策略、测试计划、测试用例规格说明、测试报告、缺陷报告等。

（2）文档审查的反馈：需求规格说明、功能规格说明、用例、设计规格说明等。

（3）信息收集与传播：开发人员、其他测试团队成员、管理者等。

所有的沟通必须是专业的、客观的，并且有效的。对于工作产品的反馈，必须保证客观性。所有的沟通应该集中在实现测试目标以及改进产品及其开发过程的质量上。测试人员需要与各种不同的角色进行沟通，包括用户、项目组成员、经理、外部测试团队和客户等。与特定相关人员的沟通必须采取不同的有效方式，例如，为开发团队设计的缺陷趋势报告，对于管理层来说可能过于详细。

下面从正确对待缺陷以及测试和开发的合作两个方面说明进行有效沟通的意义。

7.6.1　正确对待缺陷

缺陷是指可能会导致软件组件或系统无法执行其定义的功能的瑕疵，例如，错误的语句或变量定义。如果在组件或系统运行中遇到缺陷，可能会导致运行的失败。

当测试人员提交一份缺陷报告时，项目的利益干系人不应当将它看作是对产品质量的威胁。相反，应该将测试人员提交的缺陷报告看成是保证产品质量的有效手段，在测试过程中多发现并修复一个缺陷，产品的质量就可以多一分保证。因此，应当正确对待测试人员所发现的缺陷和提交的缺陷报告。

缺陷也有生命周期，一个完整的缺陷的生命周期包括：识别、调查、改正和总结。缺陷在整个生命周期里都应受到测试人员、开发人员及管理人员的高度关注，以保证能够在短时间内高效率关闭缺陷，提高软件质量，减少软件开发和维护成本。

测试人员应当尽量准确、有效地定义和描述测试过程中发现的缺陷，一个好的缺陷描述能达到事半功倍的效果，有助于缺陷的快速修复。缺陷存在放大效应，即随着时间的推移，到软件开发后期修复缺陷的成本将成倍增加。因此，当缺陷被提交之后，缺陷相关人

员应当尽快修复。

7.6.2 开发和测试的合作

首先需要明确：测试是开发的朋友，而不是开发的敌人。在实际的项目开发过程中，测试人员和开发人员有时无法有效地一起协同工作。究其原因可能有两点：一方面是因为测试人员和开发人员工作的性质不同，开发人员的工作常常可以看作是“创造性地”构建系统，而测试人员的工作在一定程度上带有破坏性；另一方面，也可能是因为管理层面的原因造成了测试人员和开发人员之间的对立。不管是由于工作性质的不同，还是由于管理层面的原因，测试人员和开发人员无法进行有效、协同地工作，都将影响产品质量的提高。

开发和测试作为一个整体都是服务于产品的，都要为产品的质量负责。从这一点来讲，开发和测试的最终目标是一致的：为客户提供高质量的产品或者服务。发现产品中的错误和缺陷是测试的重要目的之一，通过修复缺陷最终提高产品的质量，而不是去找开发人员的问题。当双方都认识到这一点时，开发和测试就有共同交流的基础。测试人员应当是开发人员的朋友，可以帮助开发人员寻找隐藏在产品中的缺陷，使得开发人员能够开发出更好的产品。

尽管有很多开发人员和测试人员也认识到了这一点，然而，在实际的开发过程中，总不免产生各种各样的摩擦。下面是为测试人员在如何处理这方面矛盾时给出的一些好的建议。

（1）一开始就采取合作而非斗争的方式，使大家明白好的软件质量是整个项目团队共同的目标。

（2）对所发现的缺陷以中立和就事论事的方式，而非批评和嘲讽的口吻与开发人员进行沟通，并且了解他人对该问题的态度。

（3）以开发人员能够理解的方式定义测试的角色和职责，帮助开发人员在产品发布之前修复缺陷，这可以降低整个项目的成本。

（4）解释自己和自己的工作，以便在疑惑的时候可以获得开发人员的帮助，降低开发人员进行不利判断的概率，并采取方式让开发人员相信，测试人员提交的缺陷报告对他们而言并非是一种威胁。

（5）缺陷报告作为测试人员和开发人员交互信息的重要媒介，测试人员在书写时应当尽量真实、公正和清楚地报告缺陷。

小结

个人的软件测试能力可以通过经验或者在不同工作领域的实践中获得。下面的每一条对增加和改进测试人员的基础知识都有帮助。

（1）软件系统的使用；

（2）领域或业务的知识；

（3）积极参与软件开发过程不同阶段的各种活动，包括分析、开发和技术支持等；

（4）积极参与软件测试过程中的各项活动；

（5）人际交往能力。

选择和招聘员工是测试经理最重要的管理职责之一。除了角色和职责要求的一些特定技能外，还有很多因素要考虑。当选择某个人加入团队时，必须考虑团队的动态性。测试团队成员技能和性格的多样性对于提高整个团队的能力非常重要。强大的测试团队能够处理各种不同复杂程度的项目，同时也能成功地处理与其他项目组成员之间的人际关系。

优秀的测试团队是做好软件测试工作的基础。测试活动的开展必须贯穿于整个软件开发生命周期，因此，测试活动应该和相关的开发活动进行协调，并统一规划。不同的组织可能会以不同的方式组织测试团队，以达到组织和项目层面的测试目标。

此外，本章还介绍了测试经理常用的激励方式，以及测试人员如何进行专业的、客观的、有效的沟通。

模拟题

51．CTAL-ATM_LO-7.2.1

TM-7.2.1(K4)使用技能评估数据表，分析团队成员在使用软件系统、领域及业务知识、系统开发领域、软件测试及人际交往技能方面的强项和弱项。

问题：

表 7-14 显示了针对用户验收测试（UAT）团队的技能矩阵，主要关注在软件开发和领域知识（出版行业）。根据每个团队成员在出版行业的领域知识、用例、软件设计和编码等方面的能力进行评估。每个软件开发领域，都从 4 个级别进行定级（无知识、了解、能理解和能创建），领域知识以高、中或低进行分类。

表 7-14 技能矩阵

技能列表	团 队 成 员				
	V	**W**	**X**	**Y**	**Z**
出版行业知识	高	低	中	高	低
用例	无知识	了解	了解	能理解	能理解
软件设计	无知识	了解	能创建	了解	能理解
编码	无知识	能创建	能理解	能理解	了解

UAT 团队决定采用类似 Java 的测试脚本语言和关键字驱动的开源测试工具。根据上面的技能矩阵，应该建议哪两个测试人员开发该测试工具框架？

选项：

A．W、X
B．V、Y
C．X、Z
D．Y、Z

解释：

A．正确。在实际脚本化中需要编码技能，而设计技能对于确保关键字框架的设计很有用。
B．不正确。他们两个都不会写代码，写脚本时需要会写代码。
C．不正确。他们两个都不会写代码，写脚本时需要会写代码。
D．不正确。他们两个都不会写代码，写脚本时需要会写代码。

分值：2 分

52．CTAL-ATM_LO-7.2.2

TM-7.2.2(K4)分析给定的团队技能评估，制定一份培训及技能培养计划。

问题：

假设你管理 6 人组成的测试团队，你已经为团队识别了三类关键技能：测试、技术和业务知识。根据每个关键技能，你对团队中每个成员进行了技能评估，评分标准是 1～5 分，其中，1 分代表技能的最低级别，而 5 分代表技能的最高级别。假设在每个类别，你得到的团队平均分数如下。

测试：3.25 分

技术：1.17 分

业务知识：3.75 分

你准备采取行动以提升团队的能力。下面哪三个选项应该包含在计划中？

选项：

A．识别最关键的特定技能存在的薄弱环节，并采取措施以改善这些薄弱环节

B．假如有招聘新人的机会，合适的应聘者应该有相关的技术能力

C．评估测试人员的人际交往技能、个性特性和沟通能力

D．邀请某个公司对所有测试人员进行 ISTQB 初级培训

E．做一次员工技能排名，并开除那些排名靠后的员工

F．尝试轮换业务知识丰富的外部人员，以支持技术能力高的人员

G．建议团队中的每个成员提高他们自己的技术能力

解释：

A．正确。需要紧迫地识别那些最可能影响有效性和效率的薄弱环节。

B．正确。应该在招聘的时候寻找平衡弱势和优势的方法。

C．正确。除了测试、技术和业务知识，软技能也是成功的关键因素。

D．不正确。已知部分或全部测试人员都已经经过认证，另外，无法得出结论：最关键技能的弱势在测试环节。

E．不正确。这是一个极端的步骤，也没有得到测试团队是失败的信息，或者看着将要失败。

F．不正确。将当前优势的技能替换为弱势的，并不是想要的解决方案。

G．不正确。放弃了管理层的职责，将指导技能成长和解决技能弱势变成了每个测试人员的问题。

分值：3 分

53．CTAL-ATM_LO-7.3.1

TM-7.3.1(K2) 针对给定的情况，讨论领导测试团队所必须具备的硬技能和软技能。

问题：

你领导 4 人组成的测试团队，负责的项目交付时间只剩 4 个星期，原测试计划显示剩余的测试的工作量是 200 人天。

有两个新的团队成员将从下周开始测试工作。

在接下来几个星期中，下面哪三个选项最好地描述了你要展示的技能？

选项：

A．处理项目进度的问题以及客户和管理层过高期望的问题

B．说服测试团队：他们是有价值的，他们的输入对团队工作量而言非常有价值

C．快速地将新员工吸收到团队里，同时提供足够的监管和支持

D．确保对所有的团队成员一视同仁，在团队内分享各种任务

E．通过积极参与测试团队的工作以显示你承诺，将处理外部事务的事情授权给其他测试团队成员

F．严格管理测试团队，只有在测试人员完成他们的所有任务后，再给他们分配新的任务

G．分析测试人员的个性特性以识别所需的新技能，以增强每个测试人员的技能储备

解释：

A．正确。这可以从下面的数据看出：根据原计划无法满足进度要求，因此需要管理该期望。

B．正确。假如团队成员感觉到他们的价值，他们更乐意更多贡献。

C．正确。因为你要引入两个新的团队成员，而时间很短，需要尽可能快地将他们吸引到团队中来。

D．不正确。确保对所有的团队成员一视同仁，在团队内分享各种任务，这并不能最好地利用不同团队成员的不同技能。

E．不正确。作为团队负责人，你应具备与团队成员不同的各种职责（以及相应的技能），因此最好还是由你来管理测试人员进行测试。

F．不正确。给测试人员更多的自由度，以及允许他们更高效地管理他们的时间，这是个更好的实践。

G．不正确。尽管分析测试人员的个性以识别所需的新技能是一个很好的实践，但是在接下来的 4 个星期中并不是最适用的技能。

分值：1 分

54．CTAL-ATM_LO-7.4.1

TM-7.4.1(K2)解释独立测试选项。

问题：

某组织为当地市场开发家庭银行软件，采用的是敏捷软件开发过程。该软件依赖于来自开源社区的外部软件组件。同时使用已有的 Web 服务，该服务在开发和集成阶段由测试桩（Stub）替代。为了在跨国银行中使用，准备国际化该软件。

为了使测试适合项目要求，下面哪个是最好的建议？

选项：

A．单元和组件集成测试由开发人员完成；系统测试在开发人员的支持下，由内部独立测试团队完成；验收测试在内部独立测试团队的支持下，由银行专家完成；国际化测试外包给外部测试专家

B．单元测试由开发人员完成；组件集成测试由内部独立的测试团队完成；系统和验收测试由银行专家完成；国际化测试外包给外部测试专家

C．单元测试由开发人员完成；系统测试由内部独立的测试团队完成；用户验收测试和国际化测试，在内部独立测试团队的支持下，由银行专家完成

D．单元和组件集成测试由开发人员完成；系统测试、用户验收测试和国际化测试，在内部独立测试团队的支持下，由银行专家完成

解释：

A．正确。开发负责技术方面的工作；内部测试人员和领域专家负责功能方面的工作；外部专家负责国际化工作。

B．不正确。只有银行专家负责系统测试：谁来测试 Web 服务？

C．不正确。没有组件集成测试；没有国际化测试知识。

D．不正确。没有国际化测试知识。

分值：1 分

55．CTAL-ATM_LO-7.5.1

TM-7.5.1(K2) 列举对测试人员动机产生积极和消极影响的因素。

问题：

近期你被任命为基于 Web 项目的测试经理，该项目当前没有满足客户的期望要求。你注意到由于测试人员郁闷导致凝聚力差，测试人员持续地离开公司去找其他的工作。

下面哪个选项最可能是**挫伤**测试人员的？

选项：

A．测试人员的奖金与交付的 Web 服务的预期质量挂钩
B．测试团队成员和 Web 开发人员之间相互尊重
C．测试人员得到了更多的责任，并可以管理他们自己的时间
D．测试人员完成的工作，管理层给予了可见的认可

解释：

A．正确。奖金与交付质量相关联是会挫伤测试人员，因为测试人员是间接地影响交付质量—在该场景中，交付无法满足客户的期望要求，因此可以设想是低质量的交付。
B．不正确。根据大纲，属于激励手段。
C．不正确。根据大纲，属于激励手段。
D．不正确。根据大纲，属于激励手段。

分值：1 分

56．CTAL-ATM_LO-7.6.1

TM-7.6.1(K2) 说明哪些因素会影响测试团队内的沟通，哪些因素会影响测试团队与干系人之间的沟通。

问题：

你是某组织的测试经理，该组织在开发自动取款机 ATM 的软件。易用性测试由内部测试团队完成。在项目开始时，上层管理人员决定从内部测试团队移交到离岸团队进行外包的功能性系统测试。易用性测试期间，内部测试团队发现存在一些严重的功能缺陷。测试报告分析之后发现，相关功能模块的功能测试用例由离岸团队设计和执行，没有发现任何可能导致阻塞的缺陷。

为了讨论缓解该状况所需的步骤/活动，下面哪个是最好的沟通建议？

选项：

A．针对功能性的系统测试相关的测试工作产品，安排时间进行评审，评审人员来自内部测试团队和离岸团队

B．将详细的缺陷报告和缺陷率发送给上层管理人员，为内部测试团队开展易用性测试争取更多的时间

C．在离岸测试团队和上层管理人员之间安排视频会议，以查找没有发现缺陷的根本原因

D．将详细的缺陷报告和缺陷率发送给离岸的测试团队，要求他们提供没有发现缺陷的根本原因

解释：

A．正确。与新的外包团队构建信任关系，处世之道和客观性很重要。而评审是建设性的手段，将内部团队的知识传授给离岸团队。

B．不正确。上层管理人员没有必要查看详细的缺陷报告。

C．不正确。上层管理人员没有必要参加根本原因分析。

D．不正确。仅靠离岸团队无法有效地发现根本原因。

分值：1 分



IGMP 需求列表

1．功能性需求

功能性需求主要包括合适性、准确性、互操作性、兼容性、安全性几个方面，如附表 1 所示。

附表 1　功能性需求

标识 ID	描　　述	优先级
SRS-IGMP-FUNC-01	用户申请加入组播组：用户端主机申请加入组播组，假如条件符合，用户将成为组播组的一个成员，可以接收组播组的数据，例如，视频点播的内容	
SRS-IGMP-FUNC-02	用户申请离开组播组：用户端主机申请离开组播组，用户将无法接收组播组的数据，例如，无法收看视频点播的内容	
SRS-IGMP-FUNC-03	用户申请离开组播组：假如用户端主机是组播组的最后一个用户，网络侧将停止向系统（IPDSAM）发送组播组数据	
SRS-IGMP-FUNC-04	组播查询处理：网络侧周期性地查询，向系统内的所有组播组发送查询数据包，假如系统内有组成员存在，则向网络侧发送应答信息	
SRS-IGMP-FUNC-05	组播查询处理：网络侧指定组查询，向系统内的指定组播组发送查询数据包，假如系统内指定组播组有成员存在，则向网络侧发送指定组播组应答信息	
SRS-IGMP-FUNC-06	组播查询处理：用户侧周期性地查询，向系统的所有组播用户发送查询数据包，假如用户侧有组成员存在，则用户侧向系统发送应答信息	
SRS-IGMP-FUNC-07	组播查询处理：用户侧指定组查询，向系统的指定组播用户发送查询数据包，假如用户侧有组成员存在，则指定组播组用户向系统发送应答信息	
SRS-IGMP-FUNC-08	组成员过期处理：系统组播组在一定时间内没有收到组成员的身份报文，该组成员身份过期，将在系统中删除，即无法收到组播组的数据	
SRS-IGMP-FUNC-09	强制用户离开组播组：假如用户的组播组权限发生变化，需要强制用户离开组播组，即中断用户的组播组业务	
SRS-IGMP-FUNC-10	强制组播源离开：假如组播源的控制权限发生变化，需要强制系统离开组播源，即中断本组播源从网络侧流向系统的业务	
SRS-IGMP-FUNC-11	发送成员报文：用户侧向系统发送组播组加入申请，假如用户是系统的第一个组播组成员，加入成功后，系统需要主动向上一级的组播路由器发送成员报文	
SRS-IGMP-FUNC-12	发送成员报文：用户侧向系统发送组播组加入申请，假如用户不是系统的第一个组播组成员，加入成功后，系统不需要主动向上一级的组播路由器发送成员报文，而是按照 SRS-IGMP-FUNC-04/05/06/07 的需求进行处理即可	

续表

标识 ID	描　述	优先级
SRS-IGMP-FUNC-13	发送成员离开报文：系统收到组播组成员离开报文，并且该用户是系统组播组中的最后一个成员，则系统向上一级组播路由器发送成员离开报文	
SRS-IGMP-FUNC-13	发送成员离开报文：假如收到可控组播模块 MCC 用户离开报文，并且该用户是系统组播组中的最后一个用户成员，则系统向上一级组播路由器发送成员离开报文	
SRS-IGMP-FUNC-14	发送成员离开报文：假如系统范围内组播组过期，则系统向上一级组播路由器发送成员离开报文	
SRS-IGMP-FUNC-15	组播源控制：用户访问的组播源是有限制的，通过访问控制列表的方式来控制用户能够访问的组播源	
SRS-IGMP-FUNC-16	基于流量的查询：系统可以提供针对每个用户的每个组播组流量进行数据统计和计费	
SRS-IGMP-FUNC-17	基于时长的查询：系统可以提供针对每个用户的每个组播组时长进行数据统计和计费	
SRS-IGMP-FUNC-18	用户接口：系统可以提供命令行 CLI 方式来配置系统的 IGMP 代理功能	
SRS-IGMP-FUNC-19	用户接口：系统可以提供图形界面 GUI 方式来配置系统的 IGMP 代理功能	
SRS-IGMP-FUNC-20	IGMP 代理功能可以从系统层面进行启动和关闭	
SRS-IGMP-FUNC-21	系统可以将 IGMP 的配置文件保存在本地闪存	
SRS-IGMP-FUNC-22	系统可以将保存在本地闪存的数据库上传到 TFTP 服务器	
SRS-IGMP-FUNC-23	系统可以通过 TFTP，将保存在 TFTP 服务器的数据库下载到系统的本地闪存，重启之后能够恢复配置，并正常工作	
SRS-IGMP-FUNC-24	系统可以设置和网络侧进行 IGMP 报文交互的路由端口（和上一级路由器进行交互 IGMP 报文交互的端口）	
SRS-IGMP-FUNC-25	IGMP 状态机相关的参数配置，符合 IGMP 相关标准 RFC 定义的默认值和取值范围。假如参数之间存在依赖关系，它们之间的关系和 RFC 标准一致	
SRS-IGMP-FUNC-26	从网络侧来的未知组播数据流，系统将直接丢弃，而不是广播每个用户端口	
SRS-IGMP-FUNC-27	从用户侧来的任何组播数据流，系统将直接丢弃，不进行任何形式的转发	
SRS-IGMP-FUNC-28	组播源可以是通过 IGMP 管理接口静态方式配置，也可以通过 IGMP 动态方式配置	
SRS-IGMP-FUNC-29	系统级别的 IGMP 处于关闭状态下，任何 IGMP 相关的报文直接丢弃，不进行任何处理或者转发	
SRS-IGMP-FUNC-30	关闭 IGMP 功能，系统将 IGMP 功能相关的统计数据全部清除，并且停止转发原来的组播业务	

2．可靠性需求

可靠性需求包括成熟性、容错性、易恢复性，如附表 2 所示。

附表 2 可靠性需求

标识 ID	描 述	优先级
SRS-IGMP-RELI-01	使用 BUILDRUN 配置文件保存配置，系统重新启动后系统读取配置文件恢复配置 并正常工作	
SRS-IGMP-RELI-02	系统能够对 IGMP 参数配置的各种异常输入，输出错误信息和告警信息，并且不影响系统已有的业务性能	
SRS-IGMP-RELI-03	系统在异常断电重启之后，能够正常恢复系统的配置，并恢复系统原来的业务	
SRS-IGMP-RELI-04	用户频繁进行组播组申请加入和离开，系统能够保持正常工作，并不影响系统已有业务	

3. 易用性需求

易用性需求如附表 3 所示。

附表 3 易用性需求

标识 ID	描 述	优先级
SRS-IGMP-USAB-01	命令行风格根据宽带命令行规范定制，提供简明的帮助信息和错误解释	
SRS-IGMP-USAB-02	命令行解释支持中英文界面	
SRS-IGMP-USAB-03	命令行输出信息和告警信息支持中英文界面	
SRS-IGMP-USAB-04	系统的 GUI 界面支持中英文界面	
SRS-IGMP-USAB-05	系统的 GUI 界面输出信息和告警信息支持中英文界面	

4. 效率需求

效率需求如附表 4 所示。

附表 4 效率需求

标识 ID	描 述	优先级
SRS-IGMP-EFFI-01	受控端口下的每个用户支持加入、发送共 4 个组播组	
SRS-IGMP-EFFI-02	同一端口下用户最大支持加入 64 个组播组	
SRS-IGMP-EFFI-03	系统最多支持 64 个组播组	
SRS-IGMP-EFFI-04	系统支持的组播数据流带宽速率在 0.5～2.5Mb/s 之间	
SRS-IGMP-EFFI-05	用户要求申请加入某个组播组，到接收到组播业务数据的时间，应该不大于 1s	
SRS-IGMP-EFFI-06	用户在不同的组播组，例如，不同的视频频道切换的时候，其切换的时间不应该超过 1s	

5. 可维护性需求

可维护性需求如附表 5 所示。

附表 5 可维护性需求

标识 ID	描 述	优先级
SRS-IGMP-MAIN-01	跟踪显示指定用户接收和发送的报文；跟踪实例由 IGMP 模块收到用户加入报文时创建，创建的 KEY 为用户 MAC 地址，同时向实例表中写入用户 VLAN ID、物理端口号，并将实例 ID 写入组播成员表中。用户离开组播组时，输出跟踪信息，并从业务跟踪对象中退出	

续表

标识 ID	描　　述	优先级
SRS-IGMP-MAIN-02	提供 IGMP 调试开关(参见 R.INTF.VRP. IGMP.013)，根据调试信息定位报文处理过程	
SRS-IGMP-MAIN-03	系统提供查看用户加入组播组和 IGMP 配置信息的命令行	
SRS-IGMP-MAIN-04	系统在接收到大负荷的 IGMP 报文的时候，例如，大于 1000 报文/秒，系统不会重启，而是将超出系统处理能力的报文丢弃	
SRS-IGMP-MAIN-05	系统默认支持的 IGMP 版本是 RFC2236，系统可以根据用户侧或者网络侧发送的 IGMP 报文版本采取系统中定义的策略，进行相关报文的应答	

6．可移植性需求

可移植性需求如附表 6 所示。

附表 6　可移植性需求

标识 ID	描　　述	优先级
SRS-IGMP-PORT-01	IGMP 软件版本可以在不同的系统硬件平台上直接运行	
SRS-IGMP-PORT-02	IGMP 软件版本升级步骤，应该简单易懂，例如，应该是黑盒级别的升级步骤描述，避免针对 IGMP 相关存储空间的操作	
SRS-IGMP-PORT-03	前面软件版本的 IGMP 数据库，通过系统移植工具的转换，IGMP 数据库可以应用到系统的当前软件版本	
SRS-IGMP-PORT-04	系统 IGMP 需要支持标准 RFC1112 IGMP version 1 和 RFC2236 IGMP version 2 和 RFC 3376 IGMP version 3	
SRS-IGMP-PORT-05	IGMP 状态机相关的参数配置，符合 IGMP 相关标准 RFC 定义的默认值和取值范围。假如参数之间存在依赖关系，它们之间的关系和 RFC 标准一致	

参 考 文 献

[1] Rex Black. Critical Testing Processes. Addison-Wesley, 2003, ISBN 0-201-74868-1.

[2] Rex Black. Managing the Testing Process, third edition. John Wiley & Sons, 2009, ISBN 0-471-22398-0.

[3] Rex Black, Erik van Veenendaal, Dorothy Graham. Foundations of Software Testing. Thomson Learning, 2011, ISBN 978-1-84480-355-2.

[4] Rick Craig, Stefan Jaskiel. Systematic Software Testing. Artech House, 2002, ISBN 1-580-53508-9.

[5] Lisa Crispin, Janet Gregory. Agile Testing: A Practical Guide for Testers and Agile Teams. Addison-Wesley, 2009, ISBN 0-321-53446-8.

[6] Gerrit de Vries, et al. TPI Next. UTN Publishers, 2009, ISBN 90-72194-97-7.

[7] Adam Goucher, Tim Riley (editors). Beautiful Testing. O'Reilly, 2009, ISBN 978-0596159818.

[8] Bob McFeeley. IDEAL: A User's Guide for Software Process Improvement. Software Engineering Institute (SEI), 1996, CMU/SEI-96-HB-001.

[9] Capers Jones. Estimating Software Costs, second edition. McGraw-Hill, 2007, ISBN 978-0071483001.

[10] Capers Jones，Olivier Bonsignour. Economics of Software Quality. Pearson, 2011, ISBN 978-0132582209.

[11] Judy McKay. Managing the Test People. Rocky Nook, 2007, ISBN 978-1933952123.

[12] John Musa. Software Reliability Engineering, second edition. Author House, 2004, ISBN 978-1418493882.

[13] D H Stamatis. Failure Mode and Effect Analysis. ASQC Quality Press, 2003, ISBN 0-873-89300.

[14] Erik van Veenendaal. The Little TMMi. UTN Publishers, 2011, ISBN 9-490-986038.

[15] Erik van Veenendaal. Practical Risk-based Testing. UTN Publishers, 2012, ISBN 978-9490986070.

[16] James Whittaker. Exploratory Testing. Addison-Wesley, 2009, ISBN 978-0321636416.

[17] Karl Wiegers. Software Requirements 2. Microsoft Press, 2003, ISBN 978-0735618794.

[18] 郑文强，马均飞. 软件测试管理. 北京：电子工业出版社，2010.

[19] Royce W. Managing the Development of Large Software Systems: Concepts and Techniques. Proc. IEEE WESCON, 1970.

[20] Andreas Spillner, Tilo Linz, Hans Schaefer. Software Testing Foundations. 北京：人民邮电出版社, 2008.

[21] Robert C Martin. 敏捷软件开发：原则、模式与实践. 邓辉，译. 北京：清华大学出版社，2003.

[22] Mo Jamshidi. System-of-Systems Engineering—a Definition. IEEE SMC，October 2005.

[23] Clifton A Ericson. Hazard Analysis Techniques for System Safety. John Wiley & Sons，2005.

[24] Qin Liu, Wenqiang Zheng, JunFei Ma. Improving Test Quality by a Test Type Analysis Based Method.

ssiri, pp.325-328, 2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement, 2009.

[25] Fenton, S Pfleeger. Software Metrics.2nd ed. PWS Publishing，1997.

[26] Len Sandler. Becoming an Extraordinary Manager: The 5 Essentials for Success. AMACOM, 2008.

[27] Steven M Bragg. Cost Accounting: Comprehensive Guideby. John Wiley & Sons,2001.

[28] [美]托马斯 • 弗里德曼. 世界是平的. 何帆，肖莹莹，译. 湖南：湖南科学技术出版社，2006.

[29] Johann Rost. The Insider's Guide to Outsourcing Risks and Rewards. Auerbach Publications，2006.

[30] Jonathan Bach. Session-Based Test Management. Software Testing and Quality Engineering magazine, 11/00.

[31] Mo Jamshidi. System of Systems Engineering. Innovations for the 21st century，A John Wilsy &Sons，2009.

[32] B Littlewood. Software Reliability: Achievement and Assessment. Blackwell Scientific Publications, November 1987.

[33] Ilene Burnstein，Taratip Suwannasart，C R Carlson. Developing a Testing Maturity Model. Crosstalk, 1996.

[34] Ilene Burnstein. Practical Software Testing: A Process-Oriented Approach. Springer，2003.

[35] I Burnstein, A Homyen, R Grom, C R Carlson. A Model for Assessing Testing Process Maturity,CrossTalk: Journal of Department of Defense Software Engineering, Vol. 11, No. 11, Nov., 1998, pp. 26-30.

[36] Ilene Burnstein. Practical Software Testing: A Process-Oriented Approach，Springer，2003.

[37] Koomen，Pol. Test Process Improvement：A Practical Step-by-Step Guide to Structured Testing (FIrst Edition). 北京：高等教育出版社，2003.

[38] SCAMPI Upgrade Team. Appraisal Requirements for CMMI（Version 1.2）. www.sei.cmu.edu/cmmi，Augest 2006.

[39] Larry R Williams. Keep'em Motivated: A Practical Guide to Motivating Employees. Marshall Cavendish，2003.

[40] A H Maslow. 动机与人格. 许金声，等译. 北京：华夏出版社，1987.

[41] CSTQB. ISTQB 软件测试高级认证大纲. www.cstqb.cn, 2012.

[42] CSTQB. ISTQB 软件测试初级认证大纲. www.cstqb.cn,2014.

[43] CSTQB. 软件测试专业术语中英文对照表(V 2.4). www.cstqb.cn,2015.

[44] IEEE 829—2008 Standard for Software Test Documentation

[45] IEEE 828—2005 Software Configuration Management Plan

[46] IEEE 1008—1987 IEEE Standard for Software Unit Testing

[47] IEEE 1028—2008 Peer Review

[48] IEEE 1059—1993 IEEE Guide for Software VeriÞcation and Validation Plan

[49] IEC 60812 failure mode and effects analysis (FMEA)

[50] IEC 61025 Fault tree analysis (FTA)

[51] IEEE 1044—1993 Incident Management

[52] IEEE Std 1061 1998 Software Quality Metrics Methodology
[53] ISO/IEC 16085—2006 Risk Management
[54] ISO/IEC TR 19760—2003 Risk Management Process
[55] ISO/IEC 9126—1—2001 Quality Model
[56] GB-T 16260—1996 信息技术 软件产品评价 质量特性及其使用指南

图书资源支持

感谢您一直以来对清华版图书的支持和爱护。为了配合本书的使用，本书提供配套的素材，有需求的用户请到清华大学出版社主页（http://www.tup.com.cn）上查询和下载，也可以拨打电话或发送电子邮件咨询。

如果您在使用本书的过程中遇到了什么问题，或者有相关图书出版计划，也请您发邮件告诉我们，以便我们更好地为您服务。

我们的联系方式：

地　　址：北京海淀区双清路学研大厦 A 座 707

邮　　编：100084

电　　话：010－62770175－4604

资源下载：http://www.tup.com.cn

电子邮件：weijj@tup.tsinghua.edu.cn

QQ：883604（请写明您的单位和姓名）

扫一扫

资源下载、样书申请

新书推荐、技术交流

用微信扫一扫右边的二维码，即可关注清华大学出版社公众号“书圈”。